网站制作·发布与维护技术实战

王 刚 编著

本书详细介绍网站制作、发布和维护的全过程
适合网站制作初学者、网站开发人员、网站站长





网站制作·发布与维护技术实战

王 刚 编著

清華大學出版社
北 京

内 容 简 介

本书针对学习开发网站的读者，详细介绍了网站从策划到编码，再到发布与管理的整个开发过程，一步步帮助读者走进网站开发的大门。

本书重点讲解网站制作主流技术，包括网页的规划、站点的建立、各种资源文件的创建、JavaScrip基本使用方法、网页结构布局、HTML5和CSS3最新设计技术、导航和表单的制作方法、在网页中插入音频和视频、简单的PHP和MySQL动态网站技术、域名和空间申请，以及网站的测试与发布。

本书适合网站开发人员、网站维护管理人员、网站设计人员、网站站长学习，也适合高校和培训学校作为相关专业的网站开发课程教学参考书使用。

图书在版编目（CIP）数据

网站制作、发布与维护技术实战 / 王刚编著. —北京：清华大学出版社，2016
ISBN 978-7-302-45279-9

I. ①网… II. ①王… III. ①网页制作工具 IV. ①TP393.092.2

中国版本图书馆CIP数据核字（2016）第248344号

责任编辑： 夏毓彦
封面设计： 王　翔
责任校对： 闫秀华
责任印制：

出版发行： 清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦A座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：
经　　销： 全国新华书店
开　　本： 190mm×260mm　　**印　张：** 21.75　　**字　数：** 557千字
版　　次： 2016年12月第1版　　**印　次：** 2016年12月第1次印刷
印　　数： 1~3500
定　　价： 59.00元

产品编号： 069959-01

前　言

随着互联网技术的迅速发展，Web 1.0和Web 2.0的时代已经离我们远去。一些互联网巨头纷纷用自己的实际行动迎接HTML 5和CSS 3技术的到来，所有主流的浏览器都已经开始支持HTML 5和CSS 3技术的很多特性。网页的代码变得越来越整洁，对搜索引擎的支持也越来越好，甚至在移动设备端，浏览器对HTML 5也提供了很好的支持。

本书针对初学者，全程介绍了网站制作、发布和维护的过程。本书共分为21章。第1~3章介绍网站的规划与准备工作，在读者对网站制作有一个整体认识之后，开始介绍如何创建站点，以及网站制作开发工具的使用方法。第4~5章逐步介绍网页中文本、图像和超链接的使用方法，因为这些内容都是每个网页中必不可少的元素。接下来介绍如何使用表格和列表展示数据，以及它们的特殊用法。第6~9章介绍CSS样式的基础知识、结合div元素布局网页结构以及制作网站导航菜单功能的方法。表单作为用户与网站交互的窗口，也进行了详细的介绍。第10~11章介绍JavaScript基础知识以及如何创建交互式网页。第12~14章介绍HTML 5和CSS 3的新功能，包括新的标签元素、音频与视频的操作方法，以及CSS 3中各种选择器的灵活使用技巧。第15章对响应式Web设计进行了简要的介绍。第16~18章开始详细介绍PHP基础知识以及MySQL数据库的使用方法，指导读者创建动态网站。第19章和20章分别介绍了域名和空间的相关知识，以及网站测试、上传和维护的方法。最后，第21章介绍了搜索引擎优化的一些知识，帮助读者提高网站的曝光率。

本书适合以下读者对象：

- 希望自己动手制作网站的初学者
- 有志于从事网站开发的专业人士
- 各种类型网站的站长
- 高校和培训机构相关专业的师生

本书配套的源代码及素材从如下网址（注意数字和字母大小写）下载：

http://pan.baidu.com/s/1bpyw6OB

如果下载有问题，请发送电子邮件至booksaga@163.com进行咨询，邮件主题设置为“网站代码”。

编者

2016年10月

目 录

第1章 规划与准备

网站建设初期的首要工作是网站策划，其重点在于对网站的设计、建设、推广和运营等各个方面进行整体的规划。不同行业的公司对于网站规划的具体要求也不一样，但一般都会确定网站的定位、类型、结构、版式和布局，以及是否使用动态布局等。



1.1 网页制作基础

网页是构成网站的基本元素，不同类型的网站由不同类型的网页组成，无论是哪种类型的网页，都是由文字、图片、动画、音频、视频等信息组成，网站的建立不能一蹴而就，需要经过详细的制作流程。

1.1.1 网页与网站

网页通常是指一个单独的页面，也就是我们在浏览器中看到的效果，但它仅限于一个网页文件。而网站由多个网页共同组成，各个网页之间通过超链接等方式连接在一起，组成网站的各个页面。网页肯定是指一个页面，即使这个页面中内嵌了其他的页面，它仍然称之为一个页面，但是网站一定是多个页面的集合，各个页面之间具有内在的关联。最简单的情况下，如果一个网站中只有一个页面，我们也可以将其称为网站。

1.1.2 网站的类型

互联网上有很多的网站，按照网站主体性质的不同，可将这些网站分为以下几类：

1. 个人网站

个人网站是以满足个人兴趣爱好为目的而开发制作的网站，这类网站往往具有十分鲜明的个人特色。例如图1.1就是一个非常独特的个人网站。

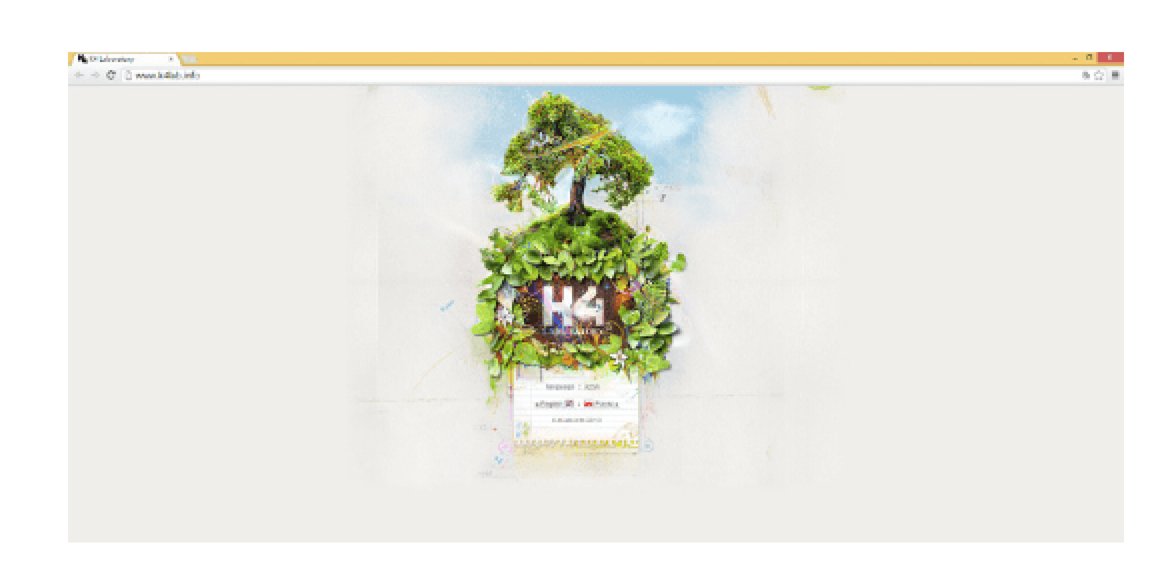


图1.1

2. 企业类网站

企业类网站是企业在互联网上进行宣传的重要窗口，用户可以在互联网上搜索关键字查找到企业的网站，从而了解到最新资讯。例如图1.2是海尔集团的网站首页。



图1.2

3. 机构类网站

机构类网站通常都是为政府或组织创建的网站，这类网站主要以形象宣传和服务为主。例如图1.3为北京大学网站首页。



图1.3

4. 娱乐休闲类网站

娱乐休闲类网站又可以分为电影网站、游戏网站、交友网站、社区论坛、手机短信网站等，这些网站为我们提供了丰富的娱乐休闲项目。随着互联网技术的高速发展，以后也许还会出现更多类型的娱乐休闲类网站。例如图1.4是国内比较有名的一个游戏网站首页。



图1.4

5. 门户类网站

互联网上的信息非常多，为了大家上网方便，有些人将这些信息进行整合、分类，然后制作成网站，这类网站综合了各个领域的资源，我们可以在这类网站上非常方便地找到自己感兴趣的资源，这样的网站就是门户类网站，国内比较知名的门户类网站有新浪、网易、搜狐等，比如图1.5是网易首页。



图1.5

6. 行业信息类网站

随着互联网的发展、网民人数的增多及网上不同兴趣群体的形成，门户网站已经明显不能满足不同群体的需要。一批能够满足某一特定领域的网上人群及其特定需要的网站应运而生，这就是行业信息类网站，比如图1.6是58同城网站首页。



图1.6

7. 购物类网站

随着互联网的普及和人们生活水平的提高，网上购物已经成为一种时尚，它正在悄然改变着人们的生活习惯。购物网站上丰富的商品、低廉的价格、优质的服务已经让越来越多的人离不开它，比如图1.7是京东商城的首页。



图1.7

1.1.3 网页的基本构成

从上面列举的各种类型的网站效果可以看出，要构建一个绚丽多彩的网页，不仅需要文本、图像和超链接，还需要灵活运用这些元素。通过合理的布局，让文本、图像和超链接融合为网站的灵魂，使用标题、段落、样式、字体颜色、背景等元素让整个页面看起来更有层次感。另外，为了达到一些特殊的效果，还可以使用动画、音频、视频等元素，甚至通过CSS样式完成一些特殊的效果，比如动画效果的导航等。

1.1.4 网站开发与制作流程

网站开发需要同时具备程序员、设计师和客户3个要素才能完成。程序员是网站开发的主力军，不但要具备深厚的程序设计功底，还需要具有良好的沟通能力。设计师负责网站整体的界面设计，不但要精通Photoshop、CorelDRAW、IIIustator、AutoCAD、3ds Max等图像处理软件，还需要熟悉Dreamweaver和Flash等制作软件，需要具备一定的审美观。客户是我们网站交付的对象，负责提供网站发布的内容、决定页面的色彩以及确定排版方式。

网站的制作流程可以分为以下几个步骤：

01 目标需求分析。网站建设初期，需要与客户确定需求，从而确定网站的内容与风格，以及应该使用的文字内容、图片、动画、音频、视频等元素，确定网站排版与各模块功能等尽量详细的需求，并形成项目计划书与客户确认。

02 制作网页。需求分析结束后，就可以开始制作网页了。此时应该首先确定网站的定位，确定网站是一个个人网站、企业类网站或者门户类网站等。只有确定了网站的类型，才能进一步确定网站的框架导航，这个过程需要程序员与设计师进行详细的讨论，包括网站各个栏目的内容，实现的方式以及如何应用按钮、图像、超链接等元素。网站排版要灵活，网站LOGO与网站色彩设置需要与客户进行沟通确认。

03 网站测试。网站开发完整后，要在本地服务器上进行系统的测试，测试通过后，通过FTP上传至网络服务器，进行网络测试。

04 上线运行。上线运行是交付网站的一个重要阶段，在上线运行期间，要密切注意网站的运行效果。

05 网站推广。网站上线后，为了提高网站的影响力，需要通过各种方式对网站进行推广，如注册搜索引擎、与其他网站交换广告条、SEO优化和媒体推广等。



1.2 网站策划

网站策划是网站建设初期的首要工作，其重点在于对网站的设计、建设、推广和运营等各个方面进行整体的策划，并提供完善的解决方案。不同的公司对于网站策划的具体要求也不一样，但一般都会确定网站的定位、类型、结构、板式与布局，以及是否使用动态布局等。

1.2.1 网站的定位

网站的定位主要考虑到网站的主题以及用户群体两个方面。目前的网站有很多，按照不同的标准，网站又可以分为很多类型，要精确把握网站的主题定位，就需要考虑当前的市场规模、用户需求以及竞争情况和潜在的对手等等，明确了这些问题，再来思考我们网站的主题定位就不再那么困难了。网站的内容与网站的用户定位息息相关，在网站策划中，只有确定了什么样的网站用户，才能精准的把握网站的内容。

1.2.2 确定网站类型

目前互联网上的网站很多，按照不同的标准可以将其划分为不同的类型，每一个网站都会有一个对应的类型。例如，前面介绍的网站按照主体性质的不同，可以分为个人网站、企业类网站、机构类网站、门户类网站、购物类网站等等。如果按照建设类型来分，可以分为自助建站和定制建站两种类型。自助建站只需要懂得如何操作电脑，就可以轻松地在几分钟内建立起一个网站，甚至还可以定制一些功能，内容比较丰富，但是不能自由迁移网站。定制建站则需要专业的网站建设团队根据你的要求，定制出一套符合你需要的系统，整个建设周期比较长，而且费用较高，但是可以自由迁移网站。

从实际情况出发，确定要建立一个什么类型的网站，需要自助建站还是定制建站，综合权衡利弊，选择适合自己的方式开始创建网站。

1.2.3 规划网站结构

规划网站结构的中心思想就是组织网站的内容和设计结构。网站定位确定后，就可以确定网站的内容，接下来就是对网站进行规划，以确保文件内容调理清晰、结构合理，这样不仅可以很好地体现设计者的意图，也可以增强网站的可维护性与可扩展性。

至于网站应该呈现什么样的内容，应该从网站设计者和网站浏览者两个角度进行考虑，从设计者的角度考虑，纵观网站的所有内容，进一步细化网站内容分类，合理组合和展示内容；从浏览者的角度考虑，根据网站风格快速定位浏览者需要的信息，并为浏览者提供最有效的信息服务。

1.2.4 确定版式与布局

网站的版式是网站的整体形象，也是网站给浏览者的第一印象。例如，同样内容的一个网站，可以是个性另类的，也可以是专业严肃的，这主要取决于站点的版面布局、浏览方式、交互性、文字、内容价格等诸多因素。

1. 页面尺寸

网站的页面尺寸会直接影响到浏览者的浏览体验。因为客观条件的影响，同一个网页在不同的分辨率下显示的效果也会有所差异，为了获得更好的浏览体验，网站设计者应该充分考虑到用户浏览器分辨率对网页的影响，使用适应浏览者的分辨率尺寸，而不是固定的页面尺寸。

2. 整体形象

网站的风格也就是网站的整体形象，通过网站的色彩、技术、文字、布局、交互方式可以概括出一个网站的个性，可以是粗犷豪放的，也可以是墨守成规的。

3. 网页布局

网页布局在网站建设中的作用越来越重要，随着技术的不断发展，浏览者已经不再满足简单的文字与图片堆叠式的布局，更多交互式的新的布局方式越来越受到人们的欢迎。

4. 网页配色

在网站策划中，对于网页配色这种非常主观的问题，需要有一个判断标准。网页设计者可以提供一些类似的网站进行分析，并征集合作人员的意见，与客户反复进行沟通确认，最终确定配色方案。

1.2.5 决定是否使用动态页面

静态页面和动态页面在网站建设中都会用到，至于哪些页面需要使用静态页面，哪些页面需要使用动态页面，这主要取决于网站的功能需求和网站内容的多少。如果网站功能比较简单，内容更新量不是很大，采用纯静态网页的方式会更简单，反之一般要采用动态网页技术来实现。



1.3 网站受众分析

网站受众分析并非凭空想象出来的，它是根据必要的市场调查，从大量的数据中对网站受众群体进行分析，进而从不同方面提升访问者的满意度。

1.3.1 必要的市场调查

市场调查可以被定义为收集和分析有关特定产品和服务的数据的过程。通过调查了解需求，比如受众群体希望网站上更多的呈现一些什么信息，什么样的信息才能吸引他们更多的关注，不同的受众群体会因为他们的年龄或社会地位而关注不同的内容，这就需要一份切合实际的市场调查，以便确定网站的目标群体。

1.3.2 如何提高访问者满意度

网站的访问者是网站存在的意义，为了提高网站访问者的满意度，可以通过以下几种方式：

（1）多途径收集用户反馈信息。
（2）深入研究用户心理。
（3）内容和服务的不断创新。
（4）竞争对手的优势分析。
（5）用户的售后服务是否到位。



1.4 搜集素材

一个完整的网站需要包含文字、图片，甚至是音频视频等很多素材，有些素材可以自己创作，有些素材可以从其他网站上下载和加工。为了便于管理这些素材，可以将搜集到的素材按照类别整理到不同的文件夹，为网站的创建做好准备。

1.4.1 网站内容需求

网站的内容由网站的定位来决定，定位明确了，网站的内容就有了方向。网站的内容并非全部必须是原创，可以是客户提供的一些信息，如公司简介、产品介绍，还可以是其他网站一些文章的引用，但需要得到许可才行。无论怎样，对于网站来说，它的内容必须能够给用户提供有价值的信息，这样才能受到用户的青睐。当然，就个人站点来说，原创的内容应该更多一些，这样也有利于搜索引擎的收录和网站的推广。

1.4.2 绘制草图

网站草图设计主要分为3个部分：网站首页、目录页以及内容页的草图设计。网站首页是网站的门面，它应该最能直观的反应出网站的主要内容。首页中的导航应该尽量简洁一些，网站中最终的内容都应该在导航中能够快速的展示。根据网站类型的不同，首页中各个栏目的内容也应该突出重点，去除一些可有可无的信息，这样才可以把用户留在自己的网站上。目录页的设计主要以简单、实用为主，根据网站的定位，向用户展示尽可能多的主题信息，这样用户才可以有更多的选择。内容页的设计需要根据不同的主题来确定，不同主题的网站在内容页的展示上有很大的差别。例如，诗歌网站的内容页应该是一篇文章，而购物网站的内容页应该是相关商品的详细介绍。

1.4.3 搜集文本与图片

文本与图片是网站承载的主要内容。文本可以通过手工录入，也可以从其他文档摘录或者通过扫描仪等设备获取，还可以从互联网上搜索。图片的搜集也有很多途径，例如通过照相设备采集、通过扫描仪扫描，或者通过互联网搜索的方式下载，如果有特殊需要，还可以

通过一些绘图软件制作图片。无论是通过哪种途径获取文本与图片，都应当尊重知识产权，不能盗用他人的文章或图片，如需引用，需要得到他人的许可。

1.4.4 组织网站内容

网站的内容大致可以分为两种，一种是网站给用户展示的信息，另一种是用户与网站的交互信息。向用户展示的信息按照内容将最主要的部分展示在最能吸引用户眼球的地方，一些特别的信息可以用粗体、颜色或一些动画效果展示，如打折促销商品可以用一个降价的图标显示。而与用户交互的信息往往会受到某些用户的青睐，这些信息可以展示在网页的侧栏，这样既可以显示当前网站备受其他人关注，还可以告诉用户其他人在这个网站上都做了些什么事情，引导用户做更多的操作。

1.4.5 建立站点地图

站点地图上面放置了网站上所有页面的链接，它是整个网站的结构图，可以帮助用户了解整个网站的内容，还可以为搜索引擎提供帮助，有利于搜索引擎抓取网页。

如果网站的内容很多，用户在浏览网站的时候可能会迷失。如果网站页面总数超过了100个，就需要挑选出最重要的页面放在站点地图中，如产品分类页面、网站内容的关键页面、访问量最大的几个页面以及帮助页面等。



选用合适的工具

Web技术发展到今天，要建立一个网站，可供我们选择的工具有很多，其中包括网页编辑器、图像处理软件和浏览器等。

1.5.1 选择合适的网页编辑器

最简单的网页编辑器就是记事本，我们可以使用记事本打开所有的网页，也可以使用记事本编写所有的网页代码，但是为了提高工作效率，以及帮助一些初学者提高编程能力，我们需要选择一款合适的文本编辑器和一款专业的HTML编辑器。

1. 选择一款文本编辑器

文本编辑器是用于编写普通文本的软件，当然也可以用于编写网页程序。目前可供选择的文本编辑器有很多，如Notepad++、PsPad、Komodo、Coda、Vim、UltraEdit、NoteTab、EditPlus等。这些文本编辑器各有各的优点，有的可以对文本的语法进行着色，有的可以自动完成拼写，根据个人的具体使用习惯，可以选择一款合适的文本编辑器。

2. 选择一款专业HTML编辑器

对于简单的页面，我们可以使用文本编辑器完成页面的制作，但是对于专业的网站来说，要提高开发效率，就需要使用专业的HTML编辑器。目前可以选择的HTML编辑器有Amaya、Dreamweaver、Frontpage、Microsoft Expression Web、CoffeeCup HTML Editor、CKEditor等，本教程将使用Dreamweaver为大家讲解网站的制作方法。

1.5.2 选择合适的图像处理软件

在制作网站时，有时候需要我们自己对一些图片进行处理，这时候就需要借助一些图像处理软件，比如Photoshop、CorelDRAW、Fireworks等。对于不同的需求，需要使用不同的图像处理软件对图像进行处理，比如要裁剪一副图片，就需要使用Photoshop。

1.5.3 选择合适的浏览器

同样一个网站在不同的浏览器中可能呈现不同的效果，这主要是因为目前可供选择的浏览器可能具有不同的引擎，他们对相同的HTML页面代码有着不同的解析方法。另外，不同的浏览器在安全性和实用性等诸多方面都会有所差别。目前主流的浏览器包括IE、Mozilla Firefox、Chrome、Opera等，为了更好的讲解HTML的相关功能，本教程将使用多种浏览器进行讲解。

第2章 创建本地站点

Dreamweaver可以用来快速编辑页面，也可以用来创建站点及其结构。在本地编辑完成的web页面可以直接在Dreamweaver中保存到站点并预览页面效果。



2.1 Dreamweaver使用基础

Dreamweaver是美国Macromedia公司开发的一款用于开发和管理网页的专业编辑器，使用该软件可以快速编写网页，并对网站进行管理。本节主要介绍Dreamweaver的一些基本使用方法。

2.1.1 Dreamweaver桌面的基本结构

Dreamweaver经过多个版本的更新，目前的最新版本是Dreamweaver CC，打开该软件后，你会看到如图2.1所示的界面。

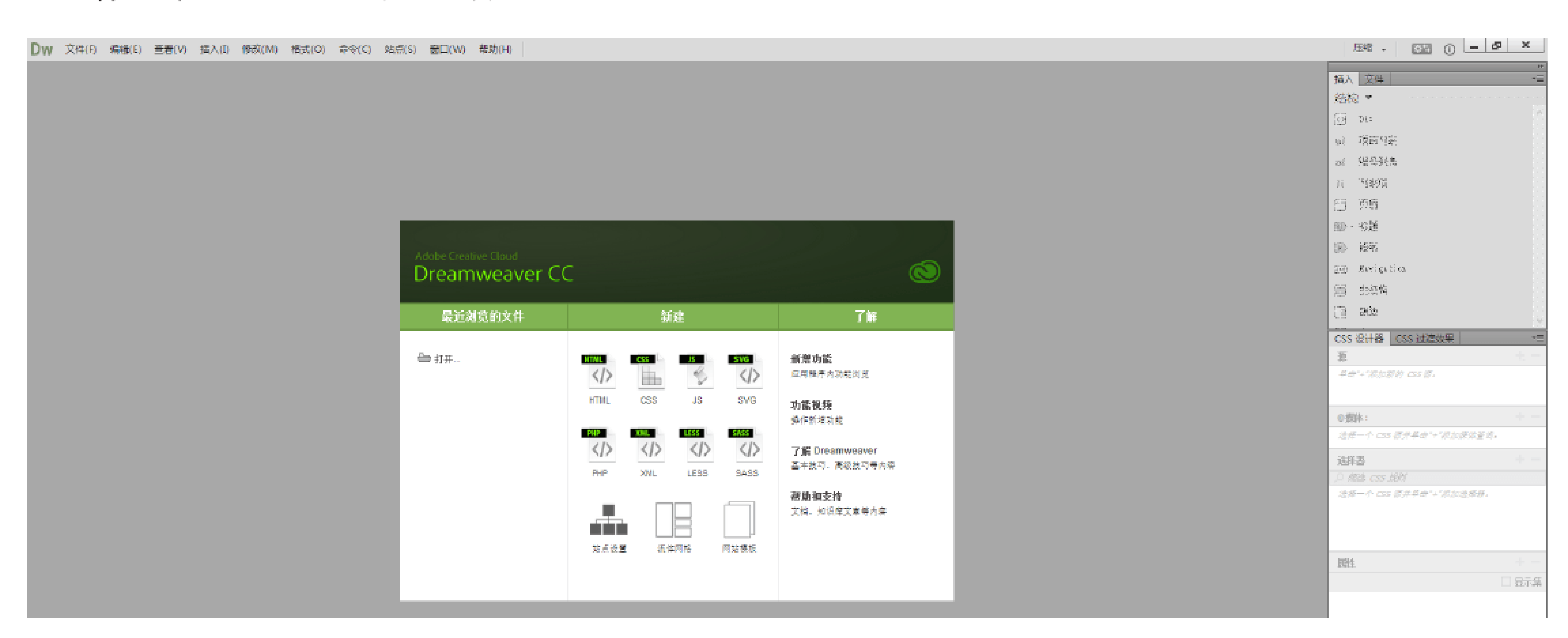


图2.1

单击该界面中的HTML图标即可创建一个HTML页面，效果如图2.2所示。

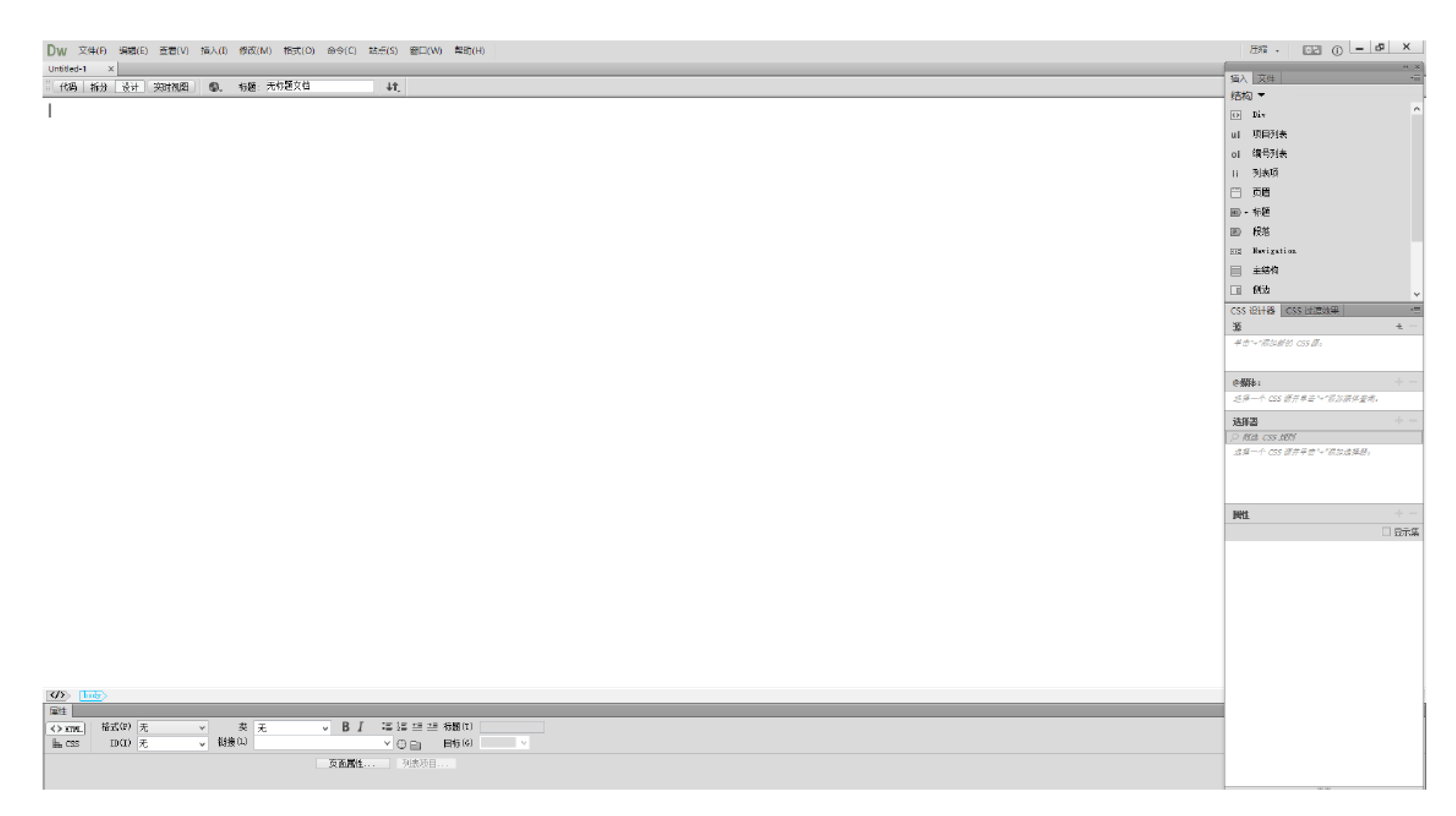


图2.2

Dreamweaver桌面的左上角是一排功能菜单按钮，通过这些菜单可以完成Dreamweaver中的所有功能操作，例如信件文件操作、代码管理、格式设置、属性设置、站点管理等。

桌面的左下角区域是属性区域，当用户在文档窗口中输入不同的内容时，属性区域会根据当前鼠标光标所在位置显示当前对象的属性。

中间的空白区域就是文档窗口，新建文档以“Untitled-”加数字序号命名，文档名称下面有3个按钮，分别用于切换代码视图、拆分视图和设计视图。例如，在代码视图中编写一段代码，这段代码用于插入一幅图片，当切换到拆分视图时，当前窗口会被分割成两个部分，一部分用于显示代码，另一部分用于显示代码呈现的效果，所以此时既可以看到代码，也可以看到插入的图片，当切换到设计视图时，用户只能看到当前页面的HTML代码效果。

桌面的右侧是一个快捷面板，在这个面版本中包含了“插入”“文件”“CSS设计器”和“CSS过滤效果”几个选项板，每个选项板中都对应有相关的操作，用于提高HTML页面的制作效率，用户还可以根据自己的需要自定义面板内容。

2.1.2 文档的基本操作

在Dreamweaver中对文档的操作主要是通过“文件”菜单来完成的，其中包括文档的新建、打开、编辑、关闭、保存、另存、导入和导出等操作。当用户启动Dreamweaver软件时，系统会自动为用户创建一个HTML文档并打开这个文档，用户可以直接在文档窗口中对当前的HTML文档进行编辑，还可以插入或粘贴从其他途径获取的HTML内容。如图2.3所示为代码视图效果。

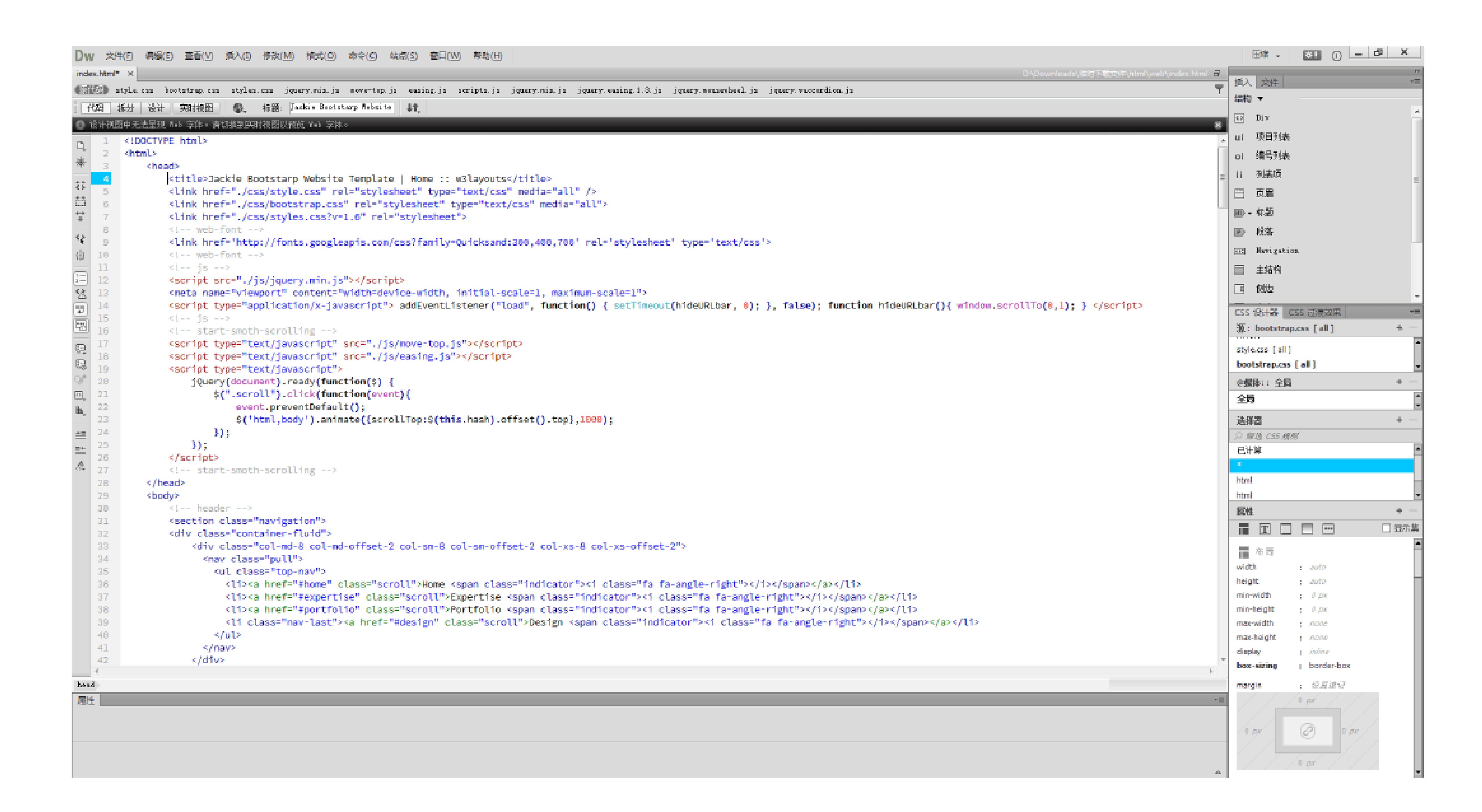


图2.3

在设计视图中，用户还可以通过“查看”菜单控制网格与标尺的显示和隐藏，如果有需要还可以设置网格的颜色和行距，以及设置标尺的单位是像素、英寸或厘米。需要注意的是，在设计视图中看到的网页效果并非是网页在浏览器中的最终效果，只有通过在浏览器中预览才能得到网页的最终效果。如图2.4所示为设计视图效果，图2.5所示为浏览器中页面的显示效果。

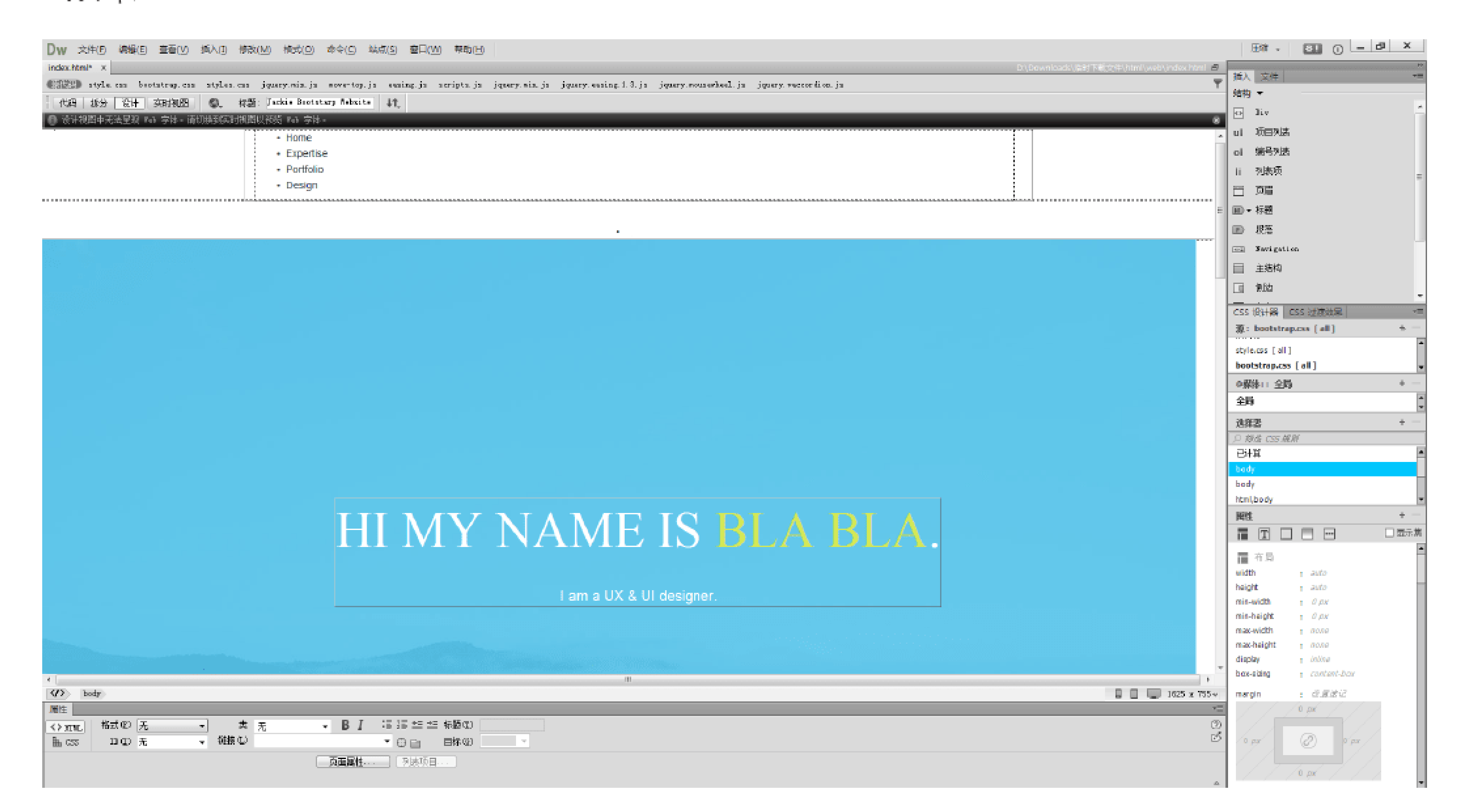


图2.4

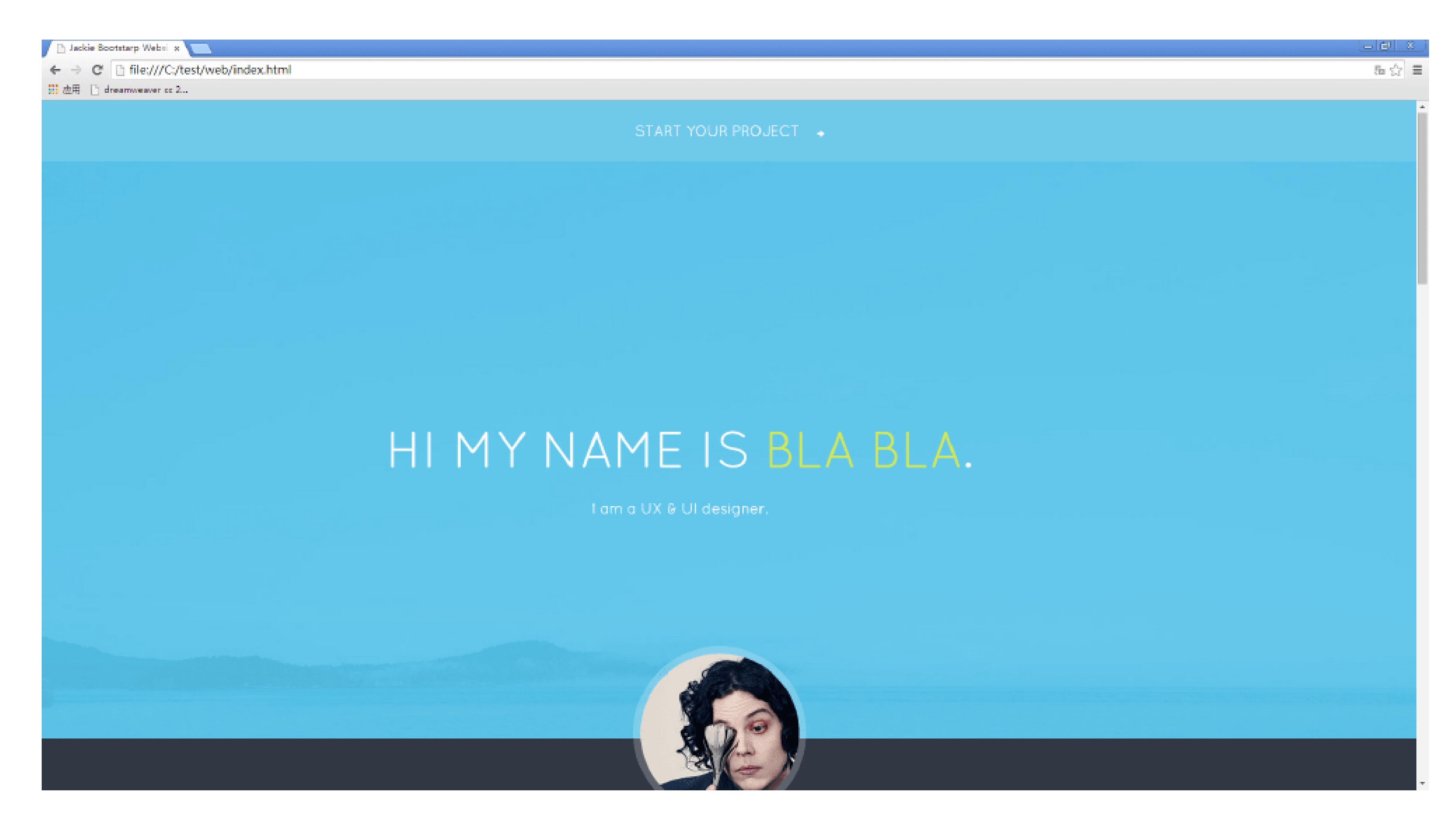


图2.5

2.1.3 源代码的格式化和净化

在代码视图中，Dreamweaver有一套自己的代码格式标准。如果HTML页面中的代码比较多，不利于阅读，可以通过“命令”菜单，选择“应用源格式”命令对当前HTML页面中的代码进行格式化操作，格式化后的代码具有对称的代码结构，非常有利用阅读。如图2.6所示为格式化操作之前的代码效果，图2.7所示为格式化操作之后的代码效果。

```
<body>
<div class="va-slice va-slice-3">
<div class="caption">
<div class="caption-grid">
<div class="caption-left">
<h3>Project Name # 12</h3>
</div>
<div class="caption-right"> </div>
<div class="clearfix"> </div>
<div class="sub-caption">
<div class="sub-caption-left"> </div>
<div class="sub-caption-right">
<p>New York</p>
</div>
<div class="clearfix"> </div>
</div>
</div>
</div>
</div>
</body>
```

图2.6

```
<body>
<div class="va-slice va-slice-3">
  <div class="caption">
    <div class="caption-grid">
      <div class="caption-left">
        <h3>Project Name # 12</h3>
      </div>
      <div class="caption-right"> </div>
      <div class="clearfix"> </div>
      <div class="sub-caption">
        <div class="sub-caption-left"> </div>
        <div class="sub-caption-right">
          <p>New York</p>
        </div>
        <div class="clearfix"> </div>
      </div>
    </div>
  </div>
</div>
</body>
```

图2.7

所谓净化HTML代码，实际上就是对HTML源代码的一种净化。很多通过网页制作工具生成的文档中存在大量无用代码和错误代码，这些代码不仅冗余，还浪费下载时间，浏览器在解析这些代码时还可能出错，所以使用Dreamweaver的“命令”菜单，选择“清理HTML”功能，不仅可以修复这些问题，还可以提高代码质量。例如，在图2.8所示的代码中，被选中的H3标记中没有任何内容，属于冗余的代码，使用源代码净化就可以非常轻松地清理类似这样的标记，而且该功能还可以指定清除特定的标记。

```
<body>
<div class="va-slice va-slice-3">
  <div class="caption">
    <div class="caption-grid">
      <div class="caption-left">
        <h3>Project Name # 12<h3></h3></h3>
      </div>
      <div class="caption-right"> </div>
      <div class="clearfix"> </div>
      <div class="sub-caption">
        <div class="sub-caption-left"> </div>
        <div class="sub-caption-right">
          <p>New York</p>
        </div>
        <div class="clearfix"> </div>
      </div>
    </div>
  </div>
</div>
</body>
```

图2.8

2.1.4 使用历史面板

Dreamweaver的历史面板中记录了在文档窗口中操作的历史，包括输入的文本、插入的对象、编辑和删除的内容等，这些操作全部保留在历史面板中，用户可以撤销一个或多个步骤，恢复文档之前的操作。



建立站点

在Dreamweaver中建立站点，是进行网站开发的一个关键步骤。这里所讲的建立站点，其实就是在Dreamweaver中定义站点、策划站点结构、部署开发环境。使用Dreamweaver创建站点的操作非常简单，详细操作步骤如下：

01 新建站点。在快捷面板中选择“文件”面板，点击该面板中的“管理站点”链接，如图2.9所示。

02 在打开的“站点管理”对话框中点击“创建站点”按钮，如图2.10所示。



图2.9

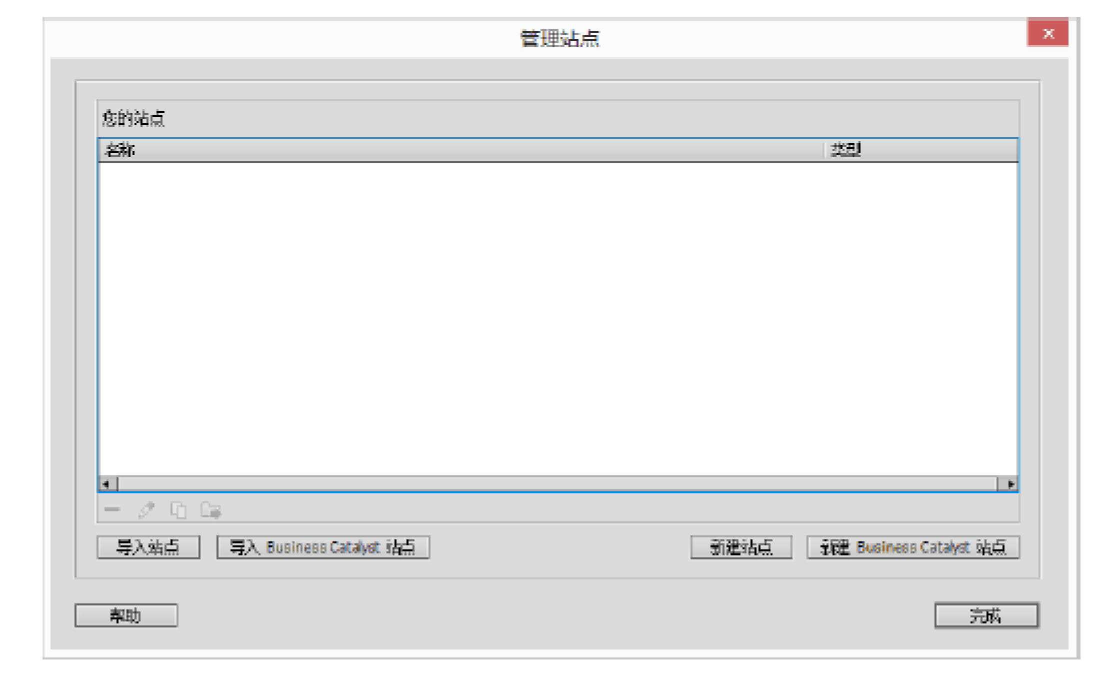


图2.10

03 在打开的“站点设置对象 我的测试站点”对话框中重新设置站点的名称和站点文件夹的路径，并保存所有的设置，如图2.11所示。

图2.11

04 在面板中选中定义的站点，右键单击站点名称，在弹出的快捷菜单中选中新建文件夹，新建“Images”“Js”和“CSS”3个文件夹，这3个文件夹将分别用于存放图片、JavaScript代码和CSS样式表，如图2.12所示。并非所有的站点结构都是这3个文件夹，根据不同的网站结构和内容，可以创建更加精细的管理结构。

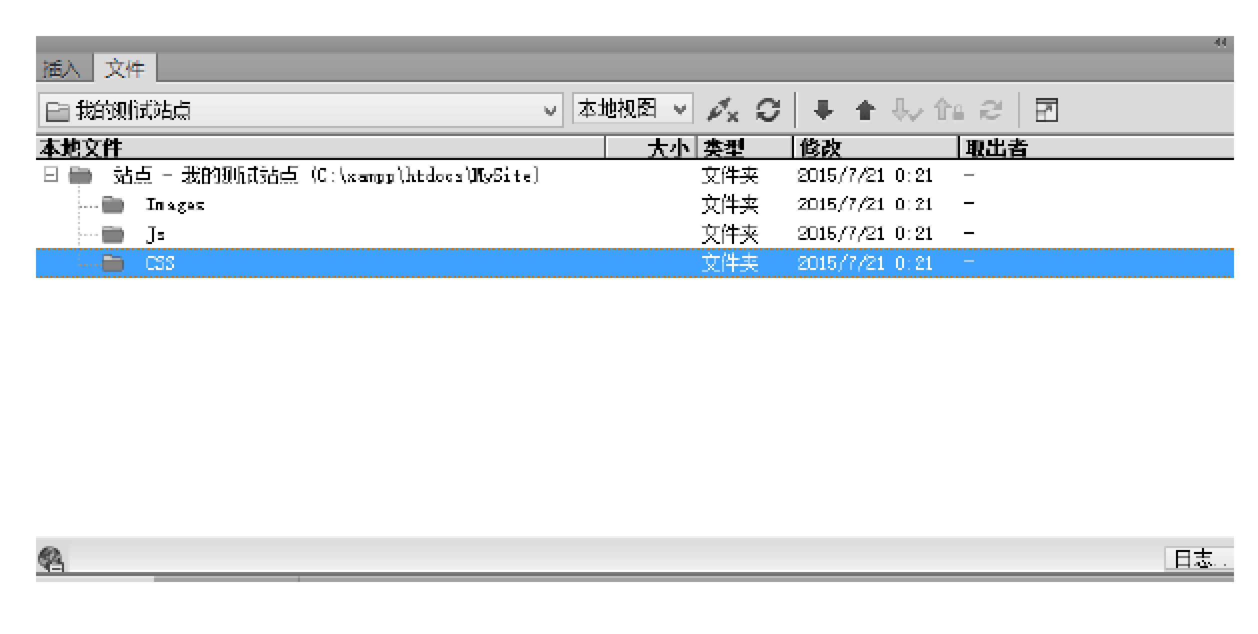


图2.12

通过以上几步简单的操作一个本地站点就创建完成了。



2.3 复制和修改站点

在菜单栏中选择“站点”→“管理站点”命令，打开“管理站点”对话框，如图2.13所示。该对话框中列举了当前所有的站点，选中一个站点名称后，单击列表下面的“复制”按钮，以当前选中的站点复制一个新的站点。如果要修改选中的站点，可以单击列表下面的“编辑”按钮，打开“站点设置对象”对话框，对站点进行编辑。

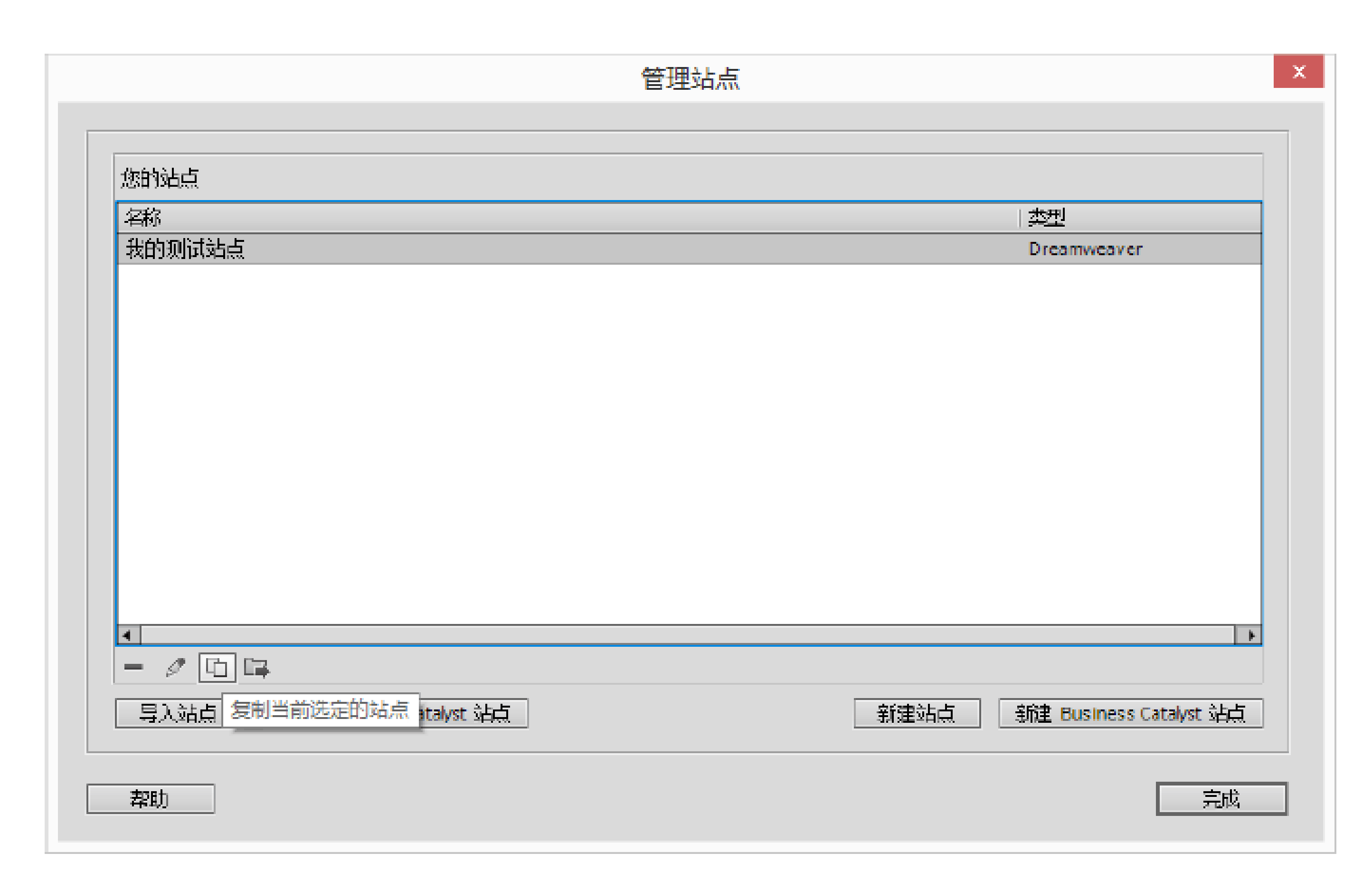


图2.14



2.4 创建第一个Web页面

任何一个复杂的网页都是由基本的网页元素构成，下面我们就来看一看如何创建基本的Web页面。

2.4.1 用记事本创建页面

对于一些简单的web页面，完全可以使用记事本完成页面的创建。例如，我们要使用记事本创建一个只显示一段文字的简单页面，就可以按照以下步骤进行操作。

01 创建一个记事本文件。
02 修改记事本文件的后缀名为html。
03 鼠标右键点击新创建的文件，在打开方式中选择以记事本打开文件。
04 在打开的文件中输入以下代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>我的页面</title>
</head>
<body>
<h2>第一个测试页面</h2>
</body>
</html>
```

05 保存文件，这样就完成了一个简单Web页面的制作。

2.4.2 用Dreamweaver创建页面

启动Dreamweaver后，在启动界面中选择“html”图标，如图2.1所示，这样就可以创建一个基本的html文档。另外，通过选择“文件”→“新建”命令，选择“新建HTML文档”也可以创建一个基本的html文档。

使用Dreamweaver创建的html文档与使用记事本创建的html文档有很大的不同。使用记事本创建的html文档，所有的html代码都需要自己输入，而使用Dreamweaver创建的html文档，系统已经默认为用户提供了基本的html文档结构，用户只需要在主体部分添加代码即可。

2.4.3 保存Web页面

在保存Web页面的时候，我们希望根据站点的结构将相关的Web页面保存在相同的文件夹中，这样当某些Web页面中的超链接发生改变的时候，系统会提示更新这些超链接，以保证站点下所有页面都能正常运行。

2.4.4 预览Web页面

在Dreamweaver的设计视图中，我们可以预览网页的效果，但是这种预览并非网页在浏览器中真正的效果，如果要预览Web页面的实际效果，可以单击文本窗口上面的“在浏览器中预览”按钮，这样就可以在浏览器中预览当前的Web页面了，用户还可以根据需要指定在哪种浏览器中进行预览，以适应更多浏览器。

第3章　在Dreamweaver中创建与使用模板

Dreamweaver作为网页开发中一款重要的工具，不仅能直观的看到网页开发的效果，而且还能极大的提高网页开发效率，其中的模板功能可以为网站提供基础模板，网站中框架类似的页面都可以通过模板来创建，这样不仅省时省力，而且还可以统一网站的风格。



创建网页模板

在创建网页模板时，可以指定模板中哪些元素是可变的，哪些元素是不可变的，可变的元素可以作为不同页面的核心内容，不可变元素可以作为整个网站中独有风格的一部分。例如，网站的logo图标和导航栏就是模板中不可变的元素，而每个页面中的主体内容则是可变的元素。这两部分内容并不是一成不变的，可以根据需要进行编辑。

在Dreamweaver中创建网页模板时，Dreamweaver会自动把模板存储在站点的本地根文件夹下的Templates子文件夹中，如果此文件夹不存在，Dreamweaver会自动创建。

3.1.1　创建空白模板

创建一个空白的模板可以按照以下步骤进行操作：

01 打开Dreamweaver，选择“文件”→“新建”命令，打开“新建文档”对话框，在该对话框中选择“空白页”，页面类型选择“HTML模板”，如图3.1所示。

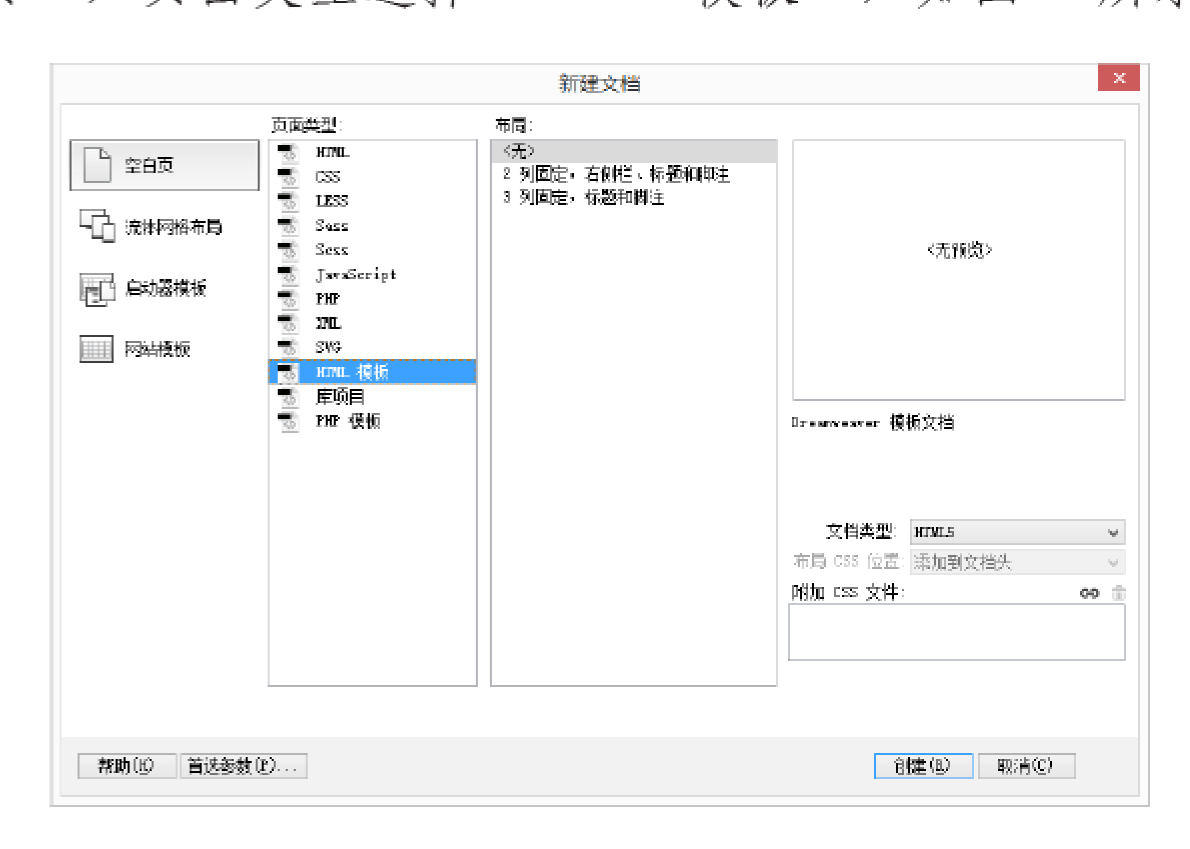


图3.1

02 单击“创建”按钮，创建一个空白的HTML模板文件。

03 选择“文件”→“保存”命令，弹出如图3.2所示的提示框，单击“确定”按钮。打开“另存模板”对话框，如图3.3所示。

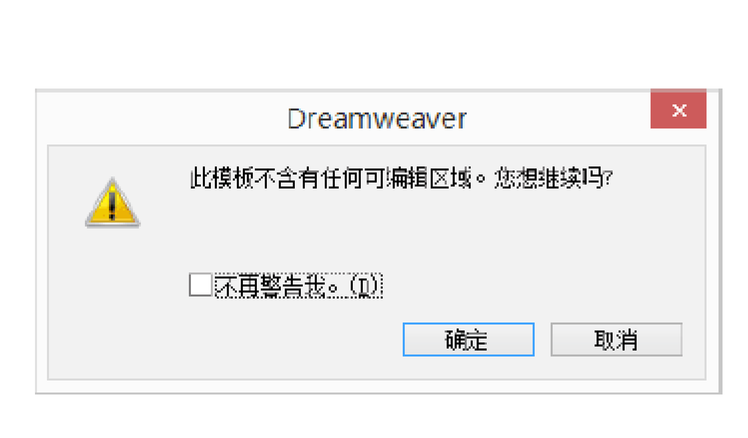


图3.2

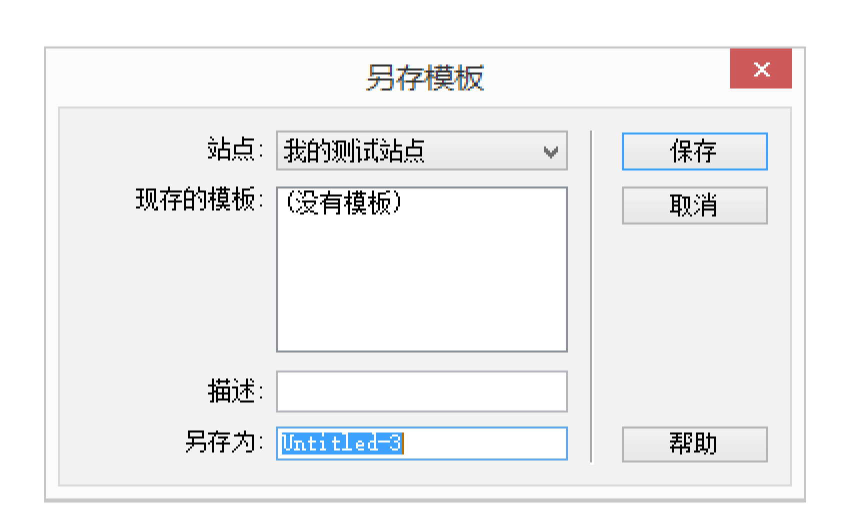


图3.3

04 设置模板保存站点和名称后，单击“保存”按钮。

这样就创建了一个空白的网页模板文件，网页模板文件的后缀名为dwt。后期还可以对这个网页模板进行编辑，设置更多的可编辑区域。

3.1.2 根据现有文档创建模板

通常情况下，我们会根据一个现有的页面文件来创建模板，具体操作步骤如下：

01 选择“文件”→“打开”命令，如图3.4所示，选择一个HTML页面，打开现有的文件。

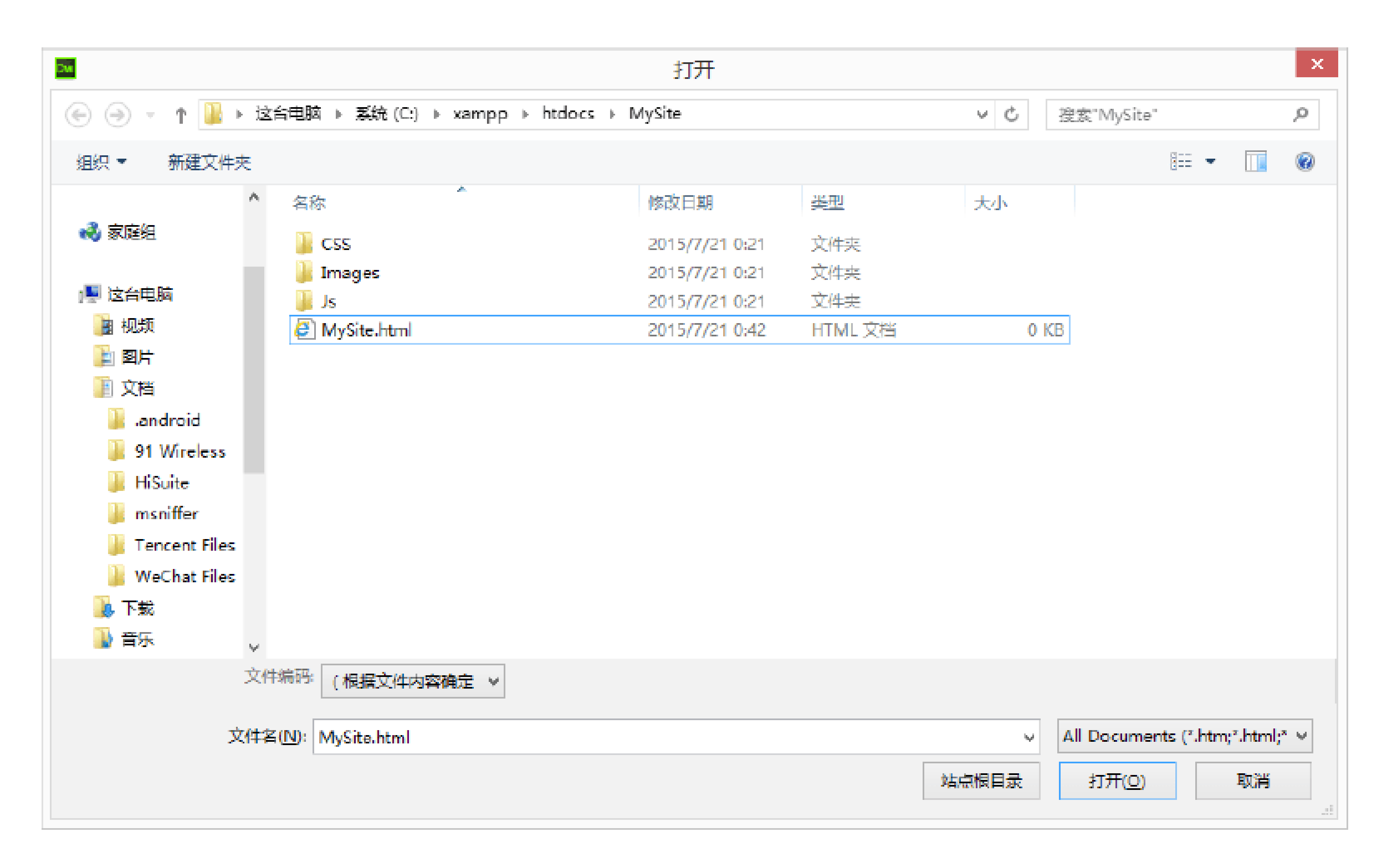


图3.4

02 选择“文件”→“另存为”命令，如图3.5所示，选择保存类型为“Template Files(*.dwt)”，设置文件名，单击“保存”按钮。

03 将光标移动到需要编辑的区域，依次选择“插入”→“模板”→“可编辑区域”命令，如图3.6所示。

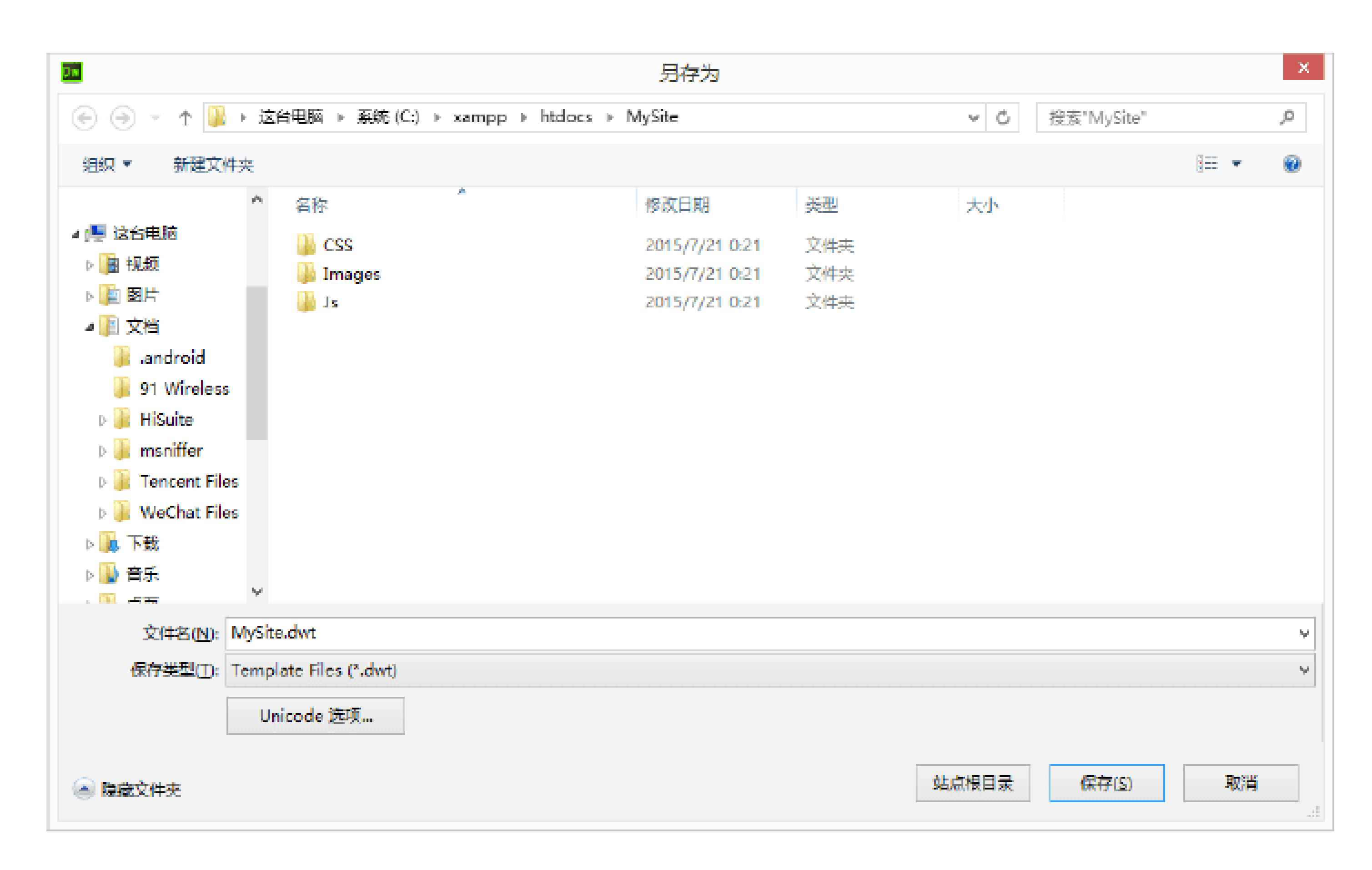


图3.5

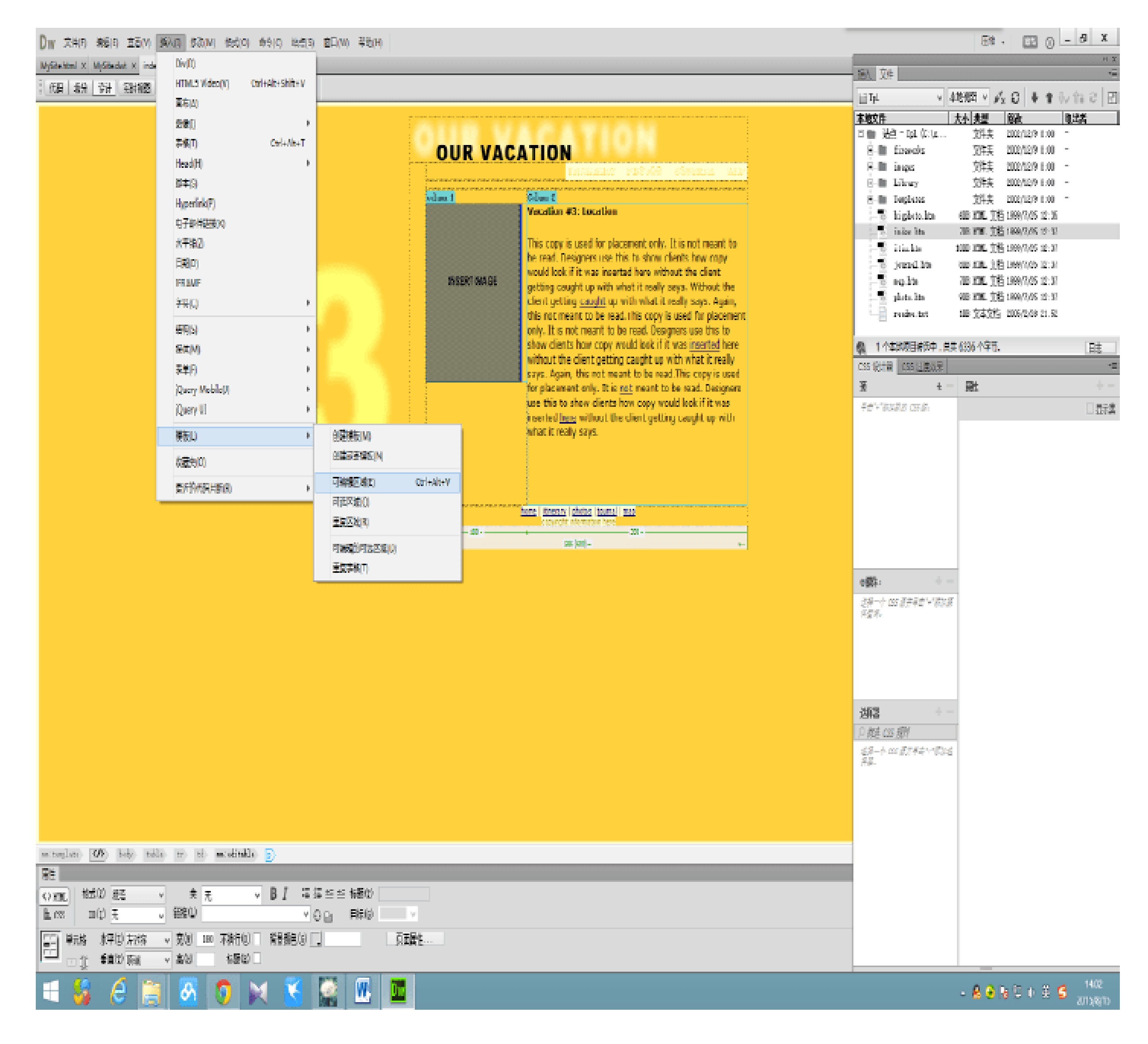


图3.6

04 重复步骤**03**的操作，设置其他可编辑区域，最终保存模板即可。



应用网页模板

创建了网页模板后，我们可以根据网页模板创建网页，具体操作步骤如下：

01 选择“文件”→“新建”命令，打开“新建文档”对话框，如图3.7所示。

02 在该对话框中左边的列表中选择“网站模板”，然后选择模板所在站点，在该站点下的模板列表中选择一个模板，最后单击“创建”按钮，这样就根据选择的模板创建了一个网页，如图3.8所示。

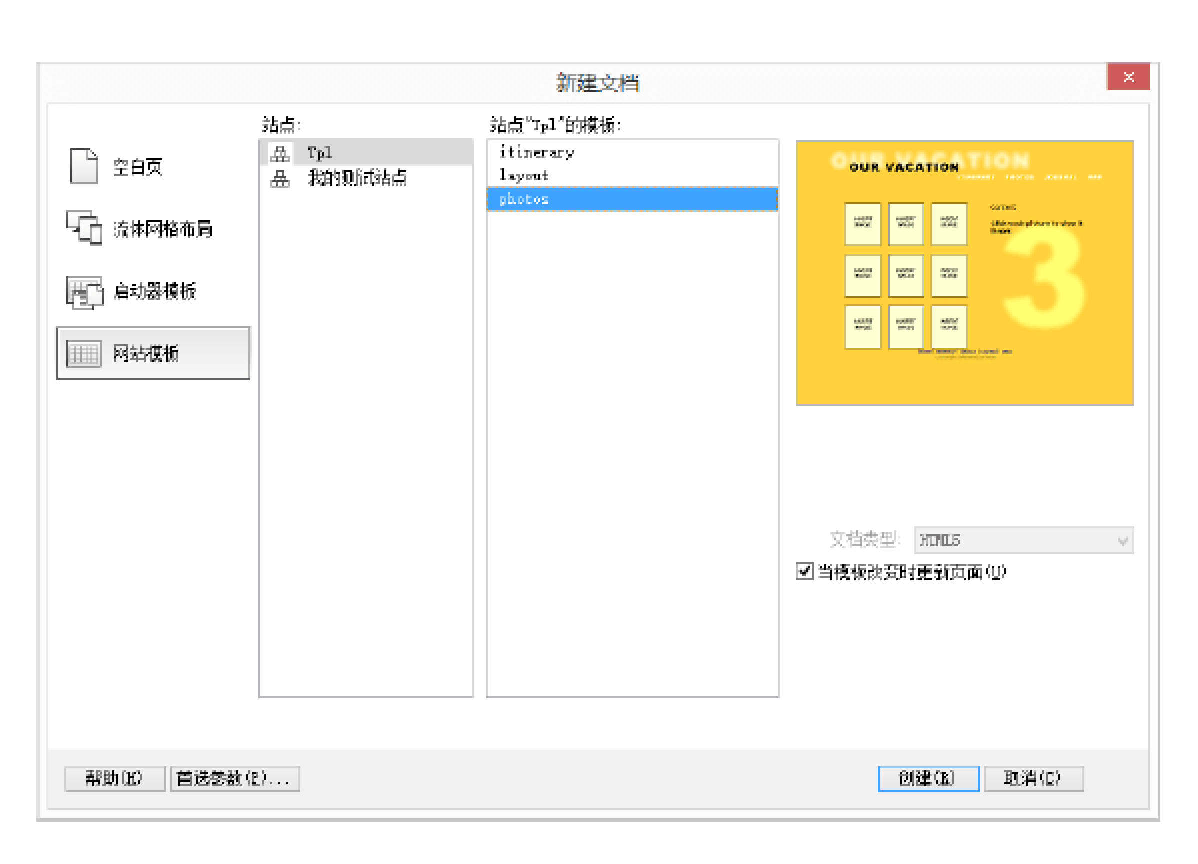


图3.7

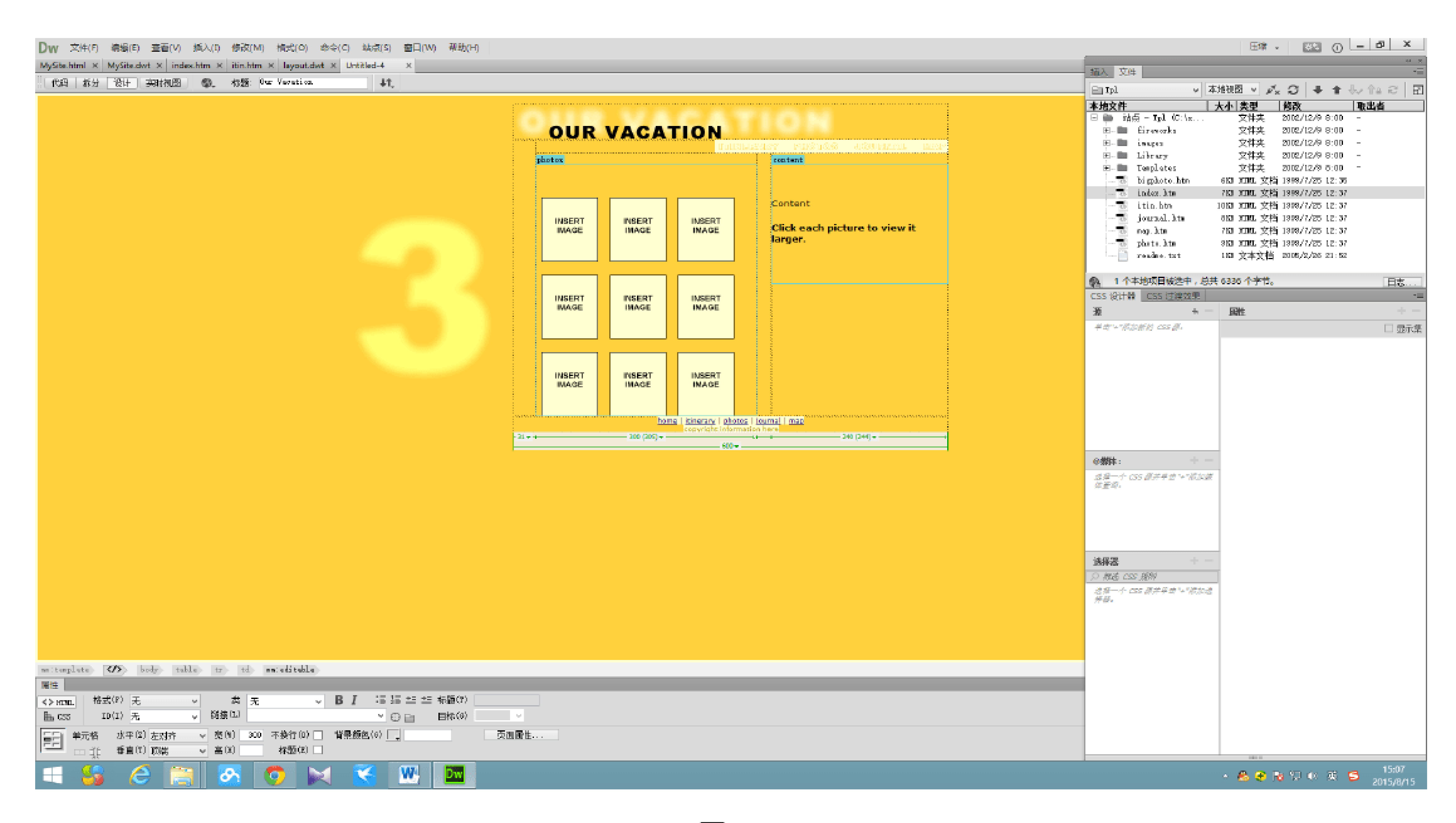


图3.8

根据模板创建的页面，当鼠标移动到可编辑区域时，可以直接输入需要编辑的内容，当鼠标移动到不可编辑区域时，鼠标会显示一个限制状态的图标。

前面说过，网页模板并不是一成不变的，我们可以根据需要对网页模板进行编辑。比如要更换一个网页模板中的logo图标，可以使用Dreamweaver打开这个网页模板，选中要更换的logo图标，并替换成新的图标。保存该模板文件时，Dreamweaver会提示用户是否更新根据此模板创建的网页，用户可以选择更新或者不更新。



3.3 简单的模板页面

本节我们将创建一个网页模板，并根据这个模板创建其他几个HTML页面，最终这几个页面会通过超链接联系在一起，详细操作步骤如下：

01 启动Dreamweaver，新建一个名为index的HTML页面。这个页面中的内容非常简单，只有一个超链接，目标地址是后面需要根据模板创建的第一个HTML页面，该页面中的代码如下所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Index</title>
<style>
a{text-decoration:none}
</style>
</head>
<body>
<h1><a href="grade_1.html">点击进入</a></h1>
</body>
</html>
```

02 选择“文件”→“另存为”命令，将步骤**01**新建的HTML页面另存为模板文件，命名为template.dwt。注意，模板文件需要保存在站点的跟目录下的Templates文件夹中，否则将不能根据站点下的模板创建文件。

03 在代码视图中为模板文件创建一个简单的页面结构，相关代码如下所示，在设计视图中的效果如图3.9所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Template</title>
<style>
body { font-family:Verdana; font-size:14px; margin:0;}
#container {margin:0 auto; width:900px;}
#sidebar { float:left; width:200px; height:500px; background:#6cf;}
#content { float:right; width:695px; height:500px; background:#cff;}
ul{list-style:none;}
a{text-decoration:none}
</style>
</head>
<body>
<div id="container">
```

```
    <div id="sidebar">
    <ul>
          <li><a href="grade_1.html">一年级</a></li>
          <li><a href="grade_2.html">二年级</a></li>
          <li><a href="grade_3.html">三年级</a></li>
     </ul>
    </div>
    <div id="content">
    <h1>诗歌朗诵</h1>
    <h3>《乡愁》</h3>
    <p>小时候<br>
    乡愁是一枚小小的邮票<br>
    我在这头<br>
    母亲在那头<br>
    长大后<br>
    乡愁是一张窄窄的船票<br>
    我在这头<br>
    新娘在那头<br>
    后来啊<br>
    乡愁是一方矮矮的坟墓<br>
    我在外头<br>
    母亲在里头<br>
    而现在<br>
    乡愁是一湾浅浅的海峡<br>
    我在这头<br>
    大陆在那头<br>
    </p>
    </div>
  </div>
  </body>
  </html>
```

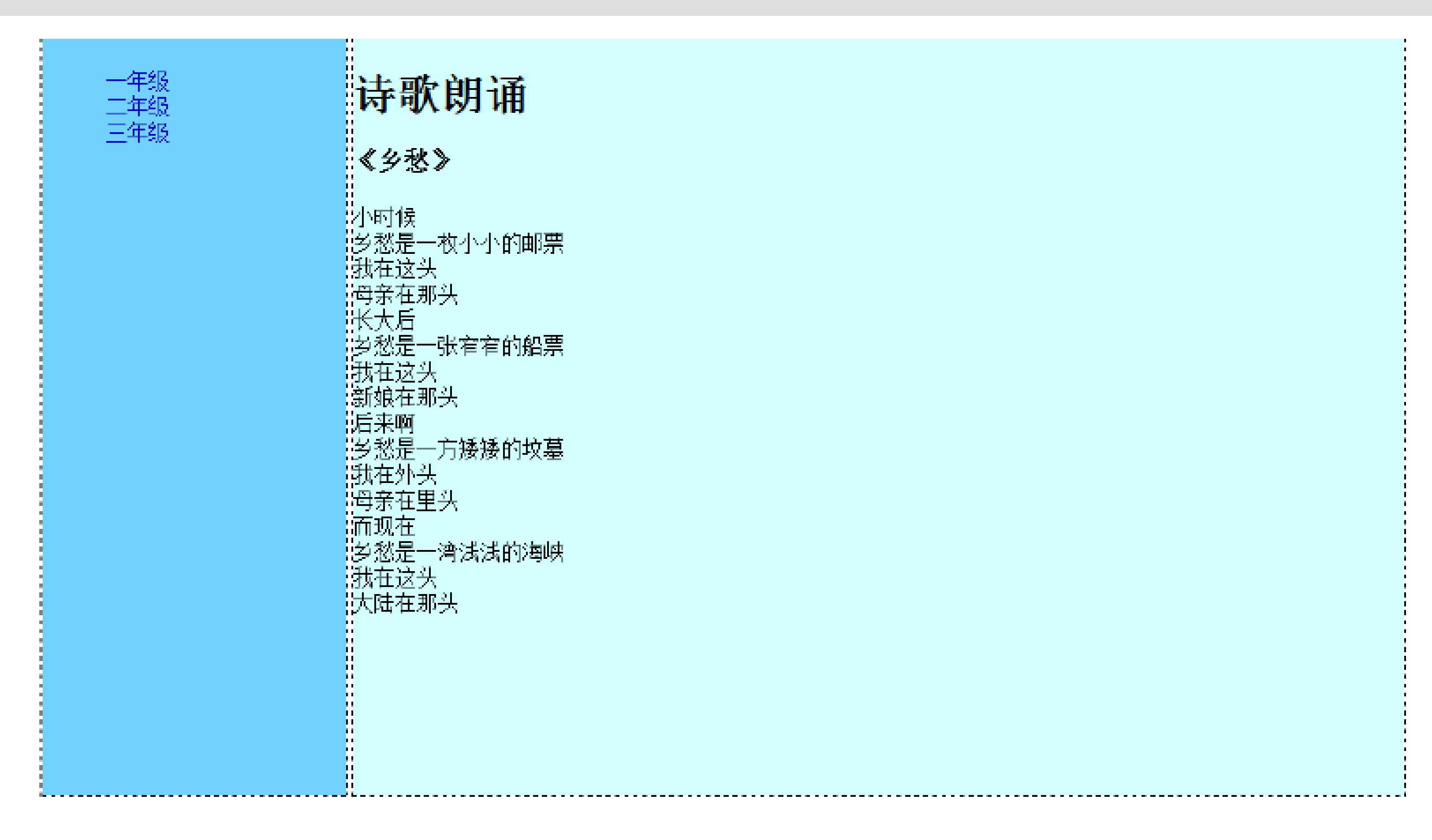


图3.9

04 用鼠标选中“诗歌朗诵”这几个字，然后按快捷键Ctrl+Alt+V，在弹出的对话框中输入可编辑区域的名称为type，然后单击“确定”按钮，如图3.10所示。

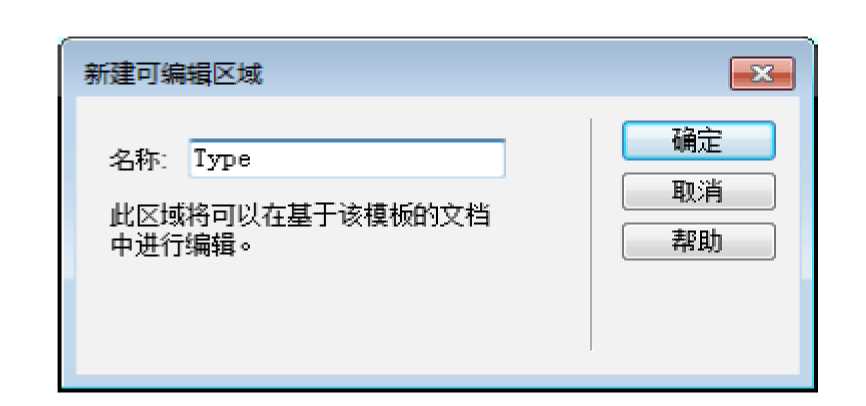


图3.10

05 重复步骤**04**，将文字“乡愁”和诗歌段落分别设置为可编辑区域，名称分别为Title和Content，保存修改后的模板文件，效果如图3.11所示。



图3.11

06 选择“新建”→“文件”命令，在弹出的“新建文档”对话框中选择“网站模板”，如图3.12所示。选中当前的站点，选择刚才创建的模板文件，单击“创建”按钮。

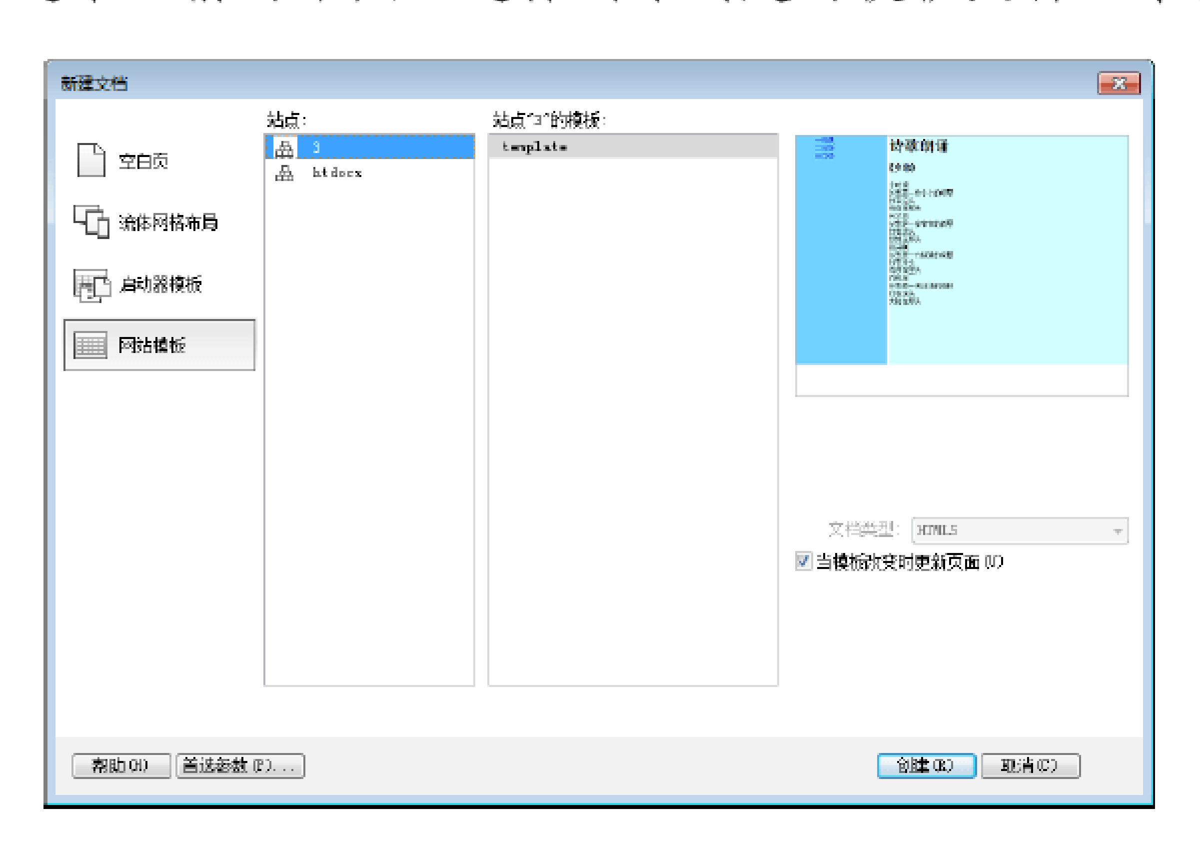


图3.12

07 根据模板创建的文件如图3.13所示，在代码视图中，不可编辑的内容显示为灰色，可编辑的内容显示为正常颜色。我们创建的第一个页面不需要改变模板中的任何内容，直接保存为grade_1.html文件即可。

```
<!doctype html>
<html><!-- InstanceBegin template="/Templates/template.dwt" codeOutsideHTMLIsLocked="false" -->
<head>
<meta charset="utf-8">
<title>Template</title>
<style>
body { font-family:Verdana; font-size:14px; margin:0;}
#container {margin:0 auto; width:900px;}
#sidebar { float:left; width:200px; height:500px; background:#6cf;}
#content { float:right; width:695px; height:500px; background:#cff;}
ul{list-style:none;}
a{text-decoration:none}
</style>
</head>

<body>
<div id="container">
  <div id="sidebar">
    <ul>
        <li><a href="file:///C|/xampp/htdocs/3/grade_1.html">一年级</a></li>
        <li><a href="file:///C|/xampp/htdocs/3/grade_2.html">二年级</a></li>
        <li><a href="file:///C|/xampp/htdocs/3/grade_3.html">三年级</a></li>
    </ul>
  </div>
  <div id="content">
  <h1><!-- InstanceBeginEditable name="Type" -->诗歌朗诵<!-- InstanceEndEditable --></h1>
  <h3>《<!-- InstanceBeginEditable name="Title" -->乡愁<!-- InstanceEndEditable -->》</h3>
  <!-- InstanceBeginEditable name="Content" -->
  <p>小时候<br>
    乡愁是一枚小小的邮票<br>
    我在这头<br>
    母亲在那头<br>
    长大后<br>
    乡愁是一张窄窄的船票<br>
    我在这头<br>
    新娘在那头<br>
    后来啊<br>
    乡愁是一方矮矮的坟墓<br>
    我在外头<br>
    母亲在里头<br>
    而现在<br>
    乡愁是一湾浅浅的海峡<br>
    我在这头<br>
    大陆在那头<br>
```

图3.13

08 根据模板再创建一个名为grade_2.html的文件，并修改可编辑区域的内容，如图3.14所示。

```
body { font-family:Verdana; font-size:14px; margin:0;}
#container {margin:0 auto; width:900px;}
#sidebar { float:left; width:200px; height:500px; background:#6cf;}
#content { float:right; width:695px; height:500px; background:#cff;}
ul{list-style:none;}
a{text-decoration:none}
</style>
</head>

<body>
<div id="container">
  <div id="sidebar">
    <ul>
        <li><a href="grade_1.html">一年级</a></li>
        <li><a href="grade_2.html">二年级</a></li>
        <li><a href="grade_3.html">三年级</a></li>
    </ul>
  </div>
  <div id="content">
  <h1><!-- InstanceBeginEditable name="Type" -->大合唱<!-- InstanceEndEditable --></h1>
  <h3>《<!-- InstanceBeginEditable name="Title" -->歌唱祖国<!-- InstanceEndEditable -->》</h3>
  <!-- InstanceBeginEditable name="Content" -->
  <p>五星红旗迎风飘扬，胜利歌声多么响亮；<br>
歌唱我们亲爱的祖国，从今走向繁荣富强。<br>
歌唱我们亲爱的祖国，从今走向繁荣富强。<br>
越过高山，越过平原，跨过奔腾的黄河长江；<br>
宽广美丽的大地，是我们亲爱的家乡，<br>
英雄的人民站起来了！我们团结友爱坚强如钢。<br>
五星红旗迎风飘扬，胜利歌声多么响亮；<br>
歌唱我们亲爱的祖国，从今走向繁荣富强。<br>
歌唱我们亲爱的祖国，从今走向繁荣富强。<br>
我们勤劳，我们勇敢，五千年历史光辉灿烂；<br>
我们战胜了一切苦难，才得到今天的解放！<br>
我们爱和平，我们爱家乡，谁敢侵犯我们就叫他灭亡！<br>
五星红旗迎风飘扬，胜利歌声多么响亮；<br>
歌唱我们亲爱的祖国，从今走向繁荣富强。<br>
歌唱我们亲爱的祖国，从今走向繁荣富强。<br>
太阳升起，万丈光芒，人民共和国正在成长；<br>
我们领袖毛泽东，指点着前进的方向。<br>
我们的生活天天向上，我们的前途万丈光芒。<br>
五星红旗迎风飘扬，胜利歌声多么响亮；<br>
歌唱我们亲爱的祖国，从今走向繁荣富强。<br>
歌唱我们亲爱的祖国，从今走向繁荣富强。<br>
  </p>
```

图3.14

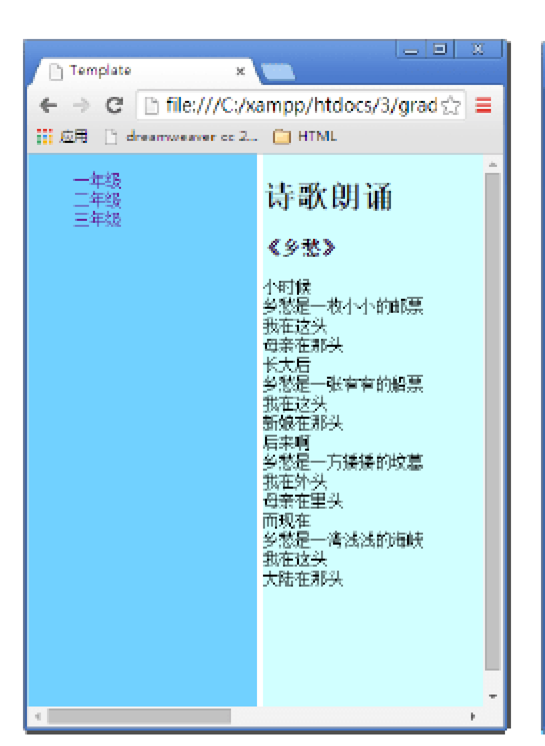
09 根据模板再创建一个名为grade_3.html的文件，并修改可编辑区域的内容，如图3.15所示。

```
body { font-family:Verdana; font-size:14px; margin:0;}
#container {margin:0 auto; width:900px;}
#sidebar { float:left; width:200px; height:500px; background:#6cf;}
#content { float:right; width:695px; height:500px; background:#cff;}
ul{list-style:none;}
a{text-decoration:none}
</style>
</head>

<body>
<div id="container">
  <div id="sidebar">
    <ul>
        <li><a href="file:///C|/xampp/htdocs/3/grade_1.html">一年级</a></li>
        <li><a href="file:///C|/xampp/htdocs/3/grade_2.html">二年级</a></li>
        <li><a href="file:///C|/xampp/htdocs/3/grade_3.html">三年级</a></li>
    </ul>
  </div>
  <div id="content">
  <h1><!-- InstanceBeginEditable name="Type" -->相声<!-- InstanceEndEditable --></h1>
  <h3>《<!-- InstanceBeginEditable name="Title" -->找地名<!-- InstanceEndEditable -->》</h3>
  <!-- InstanceBeginEditable name="Content" -->
  <p>甲：人来的不少啊<br>
乙：唉，今儿都坐满了<br>
甲：给大家介绍一下我旁边的这位，我哥们，拜把子兄弟，要说这兄弟有时候也不是什么好鸟！<br>
乙：说嘛呢？说嘛呢、谁不是什么好鸟啊？<br>
甲：你是好鸟<br>
乙：哎！嗯？有这么说话的吗？<br>
甲：我没说你，就打这么一个比方<br>
乙：没这么比的<br>
甲：人家都说嘛，宁学桃园三结义，不学瓦岗一炉香<br>
乙：哎，是有这么说的<br>
甲：像桃园三结义，刘备，关羽，张飞三人儿，一个头磕地上了，跟亲兄弟似的，回头儿，兄弟三人保唐僧西天取经去了。<br>
乙：哪儿呀这是，不对奥<br>
甲：《红楼梦》上写着呢<br>
乙：不对呀！好像是《水浒》上写的<br>
甲：唉？瞧着没，有一个比我还傻的呢<br>
乙：好嘛，你知道啊！<br>
甲：给大家介绍一下，我市非著名相生表演艺术家<br>
乙：不敢当<br>
......
  </p>
```

图3.15

10 在浏览器中浏览index.html页面，单击该页面中的超链接，在新打开的页面中单击“一年级”“二年级”和“三年级”，可分别浏览到刚才创建的页面，效果如图3.16所示。





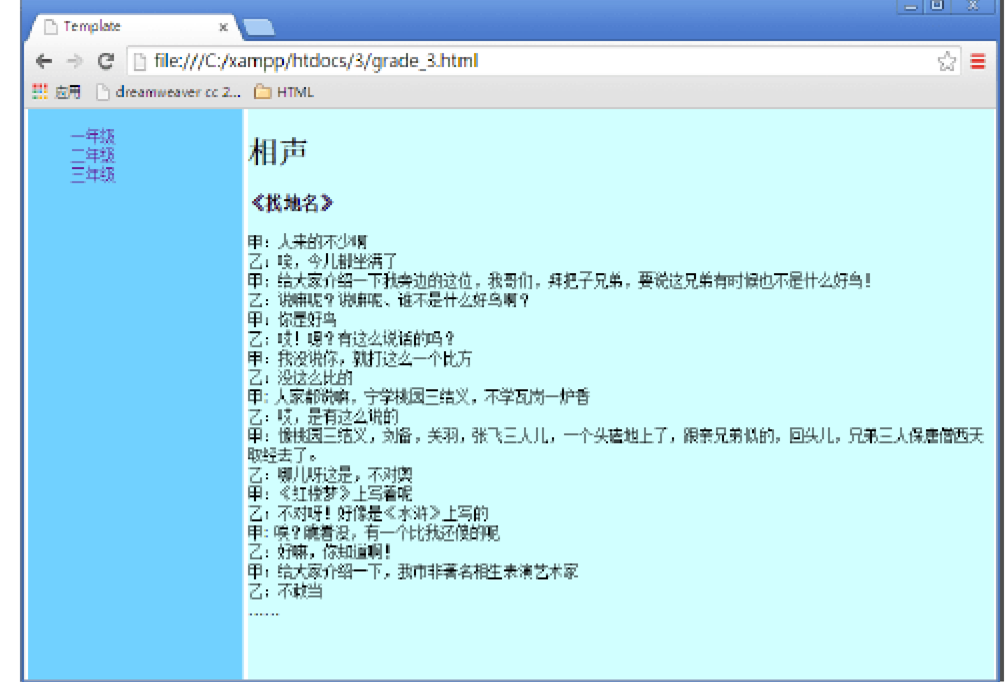


图3.16

第4章　添加文本、图像和超链接

网页中不同的元素承载着不同的内容，而文本、图像和超链接是众多元素中使用最多的3种网页元素，目前互联网上绝大多的网页都是由文本、图像和超链接这3种元素构成，本章将主要介绍这3种元素的使用方法。



4.1 认识HTML文档的结构

HTML文档由很多的标签和元素组成，自从HTML创建以来，已经经历了很多个版本，目前最新版本的HTML文档结构与早期的HTML文档机构在元素种类和书写规范上都有了很大的变化。

4.1.1 什么是HTML

HTML（Hyper Text Mark-up Language）即超文本标记语言，是WWW的描述语言，由Tim Berners-lee提出。HTML语言最初的设计目的是将多台电脑中的文本或图形联系在一起，这样只需要在任何一台电脑上打开一个文档，就可以获取多台电脑中的资源，而这些资源可以保存在网络中的任何一台电脑中。

HTML不是一种编程语言，而是一种标记语言，它通过一整套标记标签来描述网页，HTML标签可以描述文字、图形、动画、声音、表格、链接等多种形式的内容。

4.1.2 HTML版本历史

HTML版本的历史可以追溯至1993年6月，当时作为互联网工程工作小组的工作草案发布了第一版非标准的超文本标记语言，后来又分别在1995年1月和1996年1月发布了HTML 2.0和HTML 3.0，之后又于1997年12月18日由W3C推荐了HTML 4.0，于1999年12月24日由W3C推荐了HTML 4.01，直到2008年1月22日，才公布了HTML 5的第一份正式草案，目前HTML 5仍在继续完善中。

4.1.3 HTML标签

HTML标签是HTML语言中最重要的组成部分，它由两个尖括号组成。通常情况下，

HTML标签都是成对出现的，第1个出现的标签称为开始标签，第2个出现的标签称为结束标签，第2个标签会有一个结束标记，如</html>中的斜杠。某些情况下，部分HTML标签本身既是开始标签也是结束标签，如
标签。下面介绍几种常见的HTML标签。

1. 文档类型声明标签<!DOCTYPE>

DOCTYPE是文档类型（Document Type）的缩写，<!DOCTYPE>用于声明一个页面的文档类型定义（Document Type Declaration，DTD）。该标签位于文档中最前面的位置，通过确认页面的文档类型定义，可以同时确定页面使用哪种W3C规范。

2. <html>标签

<html>标签用于告诉浏览器它自身是一个HTML文档。<html>和</html>标签限定了文档的开始点和结束点，在他们之间是文档的头部和主体。

3. <head>标签

<head>标签紧跟着<html>标签，用于定义文档的头部。<head>标签中可以包含<base>、<link>、<meta>、<script>、<style>和<title>标签，由这些标签组成的元素可以引用脚本、指示浏览器在哪里使用到样式表和提供元信息等。

4. <title>标签

<title>标签是<head>标签中唯一要求包含的东西，用于在浏览器中使用标题，最常见的是显示在浏览器标签页中的标题，另外在链接列表和收藏夹书签列表中都可以看到<title>标签的身影。

5. <body>标签

<body>标签用于定义文档的主体，HTML中绝大多数标签以及元素都会在body元素中呈现，例如文本、超链接、图像、表格和列表等等。

6. <meta>标签

<meta>元素可提供有关页面的元信息（meta-information），比如针对搜索引擎和更新频度的描述和关键词。<meta>标签位于文档的头部，不包含任何内容。<meta>标签的属性定义了与文档相关联的名称/值对。<meta>标签永远位于head元素内部。

以上这些标签都是HTML页面中最基本的标签，每个HTML页面都应该具有这些标签。例如，通过Dreamweaver新建一个HTML页面，切换到代码视图，可以看到Dreamweaver已经为我们自动创建了这些最基本的标签。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>无标题文档</title>
</head>
<body>
</body>
</html>
```

在实际的项目中，网页中不仅需要具有这些基本标签，根据网站设计以及网站推广的需要，部分标签还会重复使用。例如，以下是新浪某博客网页中的部分代码，有兴趣的同学还可以通过使用浏览器查看网页源码的方式浏览到其他网页的HTML标签。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>猪肉白菜包，一分钟学会快速发面的技巧_厨恋恋_新浪博客</title>
<meta name="keywords" content="猪肉白菜包，一分钟学会快速发面的技巧_厨恋恋_新浪博客,厨恋恋,厨恋恋,美食,面点,主食,包子" />
<meta name="description" content="猪肉白菜包，一分钟学会快速发面的技巧_厨恋恋_新浪博客,厨恋恋," />
<meta http-equiv="X-UA-Compatible" content="IE=EmulateIE7" />
<meta http-equiv="mobile-agent" content="format=html5; url=http://blog.sina.cn/dpool/blog/s/blog_4ee9cd560102wp8l.html?vt=4">
<meta http-equiv="mobile-agent" content="format=wml; url=http://blog.sina.cn/dpool/blog/ArtRead.php?nid=4ee9cd560102wp8l&vt=1">
<!-[if lte IE 6]>
<script type="text/javascript">
try{
document.execCommand("BackgroundImageCache", false, true);
}catch(e){}
</script>
<![endif]->
<script type="text/javascript">
window.staticTime=new Date().getTime();
</script>
<link rel="pingback" href="http://upload.move.blog.sina.com.cn/blog_rebuild/blog/xmlrpc.php" />
<link rel="EditURI" type="application/rsd+xml" title="RSD" href="http://upload.move.blog.sina.com.cn/blog_rebuild/blog/xmlrpc.php?rsd" />
<link href="http://blog.sina.com.cn/blog_rebuild/blog/wlwmanifest.xml" type="application/wlwmanifest+xml" rel="wlwmanifest" />
<link rel="alternate" type="application/rss+xml" href="http://blog.sina.com.cn/rss/1323945302.xml" title="RSS" />
<link href="http://simg.sinajs.cn/blog7style/css/conf/blog/article.css" type="text/css" rel="stylesheet" /><link href="http://simg.sinajs.cn/blog7style/css/common/common.css" type="text/css" rel="stylesheet" /><link href="http://simg.sinajs.cn/blog7style/css/blog/blog.css" type="text/css" rel="stylesheet" /><link href="http://simg.sinajs.cn/blog7style/css/module/common/blog.css" type="text/css" rel="stylesheet" /><style id="tplstyle" type="text/css">@charset "utf-8";@import url("http://simg.sinajs.cn/blog7newtpl/css/8/8_14/t.css");
</style>
<style id="positionstyle"  type="text/css">
.sinabloghead .blogtoparea{ left:128px;top:39%;}
.sinabloghead .blognav{ left:115px;top:67%;}
</style>
```

4.1.4 HTML元素

HTML元素是指从开始标签到结束标签的所有代码。大部分的HTML元素都有开始标签和结束标签，开始标签和结束标签之间的内容称为元素的内容。元素的内容可以为空，在开始标签中使用斜杠结束标签，大多数的HTML元素都拥有属性。

例如下面这段代码：

```
<body>
<h2>列表</h2>
<hr/>
<ul>
    <li>项目</li>
    <li>项目</li>
    <li>项目</li>
</ul>
</body>
```

这段代码由<body>、<h2>、<hr/>、<ul>和<li>这些标签组成，body元素作为主体，其他的元素都嵌套在body元素开始标签和结束标签的内部，作为body元素的内容。h2元素表示元素的标题。<hr/>元素的内容为空，用于创建一条水平线，所以在开始标签中直接使用结束标签。ul和li元素用于表示列表，分别由对应的开始标签和结束标签组成，li元素作为ul元素的内容，而文字“项目”作为li元素的内容。

HTML元素有很多，在本书的讲解中，我们将逐步介绍一些常用元素的使用方法。HTML元素按功能类别排列，可以分为以下几种。

（1）基础的元素，表4.1列出了一些常用的基础元素。

表4.1

标签	描述
<!DOCTYPE>	定义文档类型
<html>	定义 HTML 文档
<title>	定义文档的标题
<body>	定义文档的主体
<h1> to <h6>	定义 HTML 标题
<p>	定义段落
 	定义简单的折行
<hr>	定义水平线
<!--...-->	定义注释

（2）格式相关的元素，表4.2列出了一些常用的格式相关的元素。

表4.2

标签	描述
<acronym>	定义只取首字母的缩写
<abbr>	定义缩写
<address>	定义文档作者或拥有者的联系信息
<b>	定义粗体文本
<bdi>	定义文本的方向，使其脱离周围文本的方向设置
<bdo>	定义文字方向
<big>	定义大号文本
<blockquote>	定义长的引用
<center>	定义居中文本（不赞成使用）
<cite>	定义引用(citation)
<code>	定义计算机代码文本
<del>	定义被删除文本
<dfn>	定义定义项目
<em>	定义强调文本
<font>	定义文本的字体、尺寸和颜色（不赞成使用）
<i>	定义斜体文本
<ins>	定义被插入文本
<kbd>	定义键盘文本
<mark>	定义有记号的文本
<meter>	定义预定义范围内的度量
<pre>	定义预格式文本
<progress>	定义任何类型的任务的进度
<q>	定义短的引用
<rp>	定义若浏览器不支持 ruby 元素显示的内容
<rt>	定义 ruby 注释的解释
<ruby>	定义 ruby 注释
<s>	定义加删除线的文本（不赞成使用）
<samp>	定义计算机代码样本
<small>	定义小号文本
<strike>	定义加删除线的文本（不赞成使用）
<strong>	定义语气更为强烈的强调文本
<sup>	定义上标文本
<sub>	定义下标文本
<time>	定义日期/时间
<tt>	定义打字机文本
<u>	定义下划线文本（不赞成使用）
<var>	定义文本的变量部分
<wbr>	定义视频

（3）表单相关的元素，表4.3列出了一些常用的表单相关的元素。

表4.3

标签	描述
<form>	定义供用户输入的 HTML 表单
<input>	定义输入控件
<textarea>	定义多行的文本输入控件
<button>	定义按钮
<select>	定义选择列表（下拉列表）
<optgroup>	定义选择列表中相关选项的组合
<option>	定义选择列表中的选项
<label>	定义 input 元素的标注
<fieldset>	定义围绕表单中元素的边框
<legend>	定义 fieldset 元素的标题
<isindex>	定义与文档相关的可搜索索引（不赞成使用）
<datalist>	定义下拉列表
<keygen>	定义生成密钥
<output>	定义输出的一些类型

（4）框架相关的元素，表4.4列出了一些常用的框架相关的元素。

表4.4

标签	描述
<frame>	定义框架集的窗口或框架
<frameset>	定义框架集
<noframes>	定义针对不支持框架的用户的替代内容
<iframe>	定义内联框架

（5）图像相关的元素，表4.5列出了一些常用的图像相关的元素。

表4.5

标签	描述
<img>	定义图像
<map>	定义图像映射
<area>	定义图像地图内部的区域
<canvas>	定义图形
<figcaption>	定义 figure 元素的标题
<figure>	定义媒介内容的分组，以及它们的标题

（6）音视频相关的元素，表4.6列出了一些常用的音视频相关的元素。

表4.6

标签	描述
<audio>	定义声音内容。
<source>	定义媒介源。
<track>	定义用在媒体播放器中的文本轨道。
<video>	定义视频。

（7）链接相关的元素，表4.7列出了一些常用的链接相关的元素。

表4.7

标签	描述
<a>	定义锚
<link>	定义文档与外部资源的关系
<nav>	定义导航链接

（8）列表相关的元素，表4.8列出了一些常用的列表相关的元素。

表4.8

标签	描述
	定义无序列表
	定义有序列表
	定义列表的项目
<dir>	定义目录列表（不赞成使用）
<dl>	定义定义列表
<dt>	定义定义列表中的项目
<dd>	定义定义列表中项目的描述
<menu>	定义命令的菜单/列表
<menuitem>	定义用户可以从弹出菜单调用的命令/菜单项目
<command>	定义命令按钮

（9）表格相关的元素，表4.9列出了一些常用的表格相关的元素。

表4.9

标签	描述
<table>	定义表格
<caption>	定义表格标题
<th>	定义表格中的表头单元格
<tr>	定义表格中的行
<td>	定义表格中的单元
<thead>	定义表格中的表头内容
<tbody>	定义表格中的主体内容
<tfoot>	定义表格中的表注内容（脚注）
<col>	定义表格中一个或多个列的属性值
<colgroup>	定义表格中供格式化的列组

（10）样式/章节相关的元素，表4.10列出了一些常用的样式/章节相关的元素。

表4.10

标签	描述
<style>	定义文档的样式信息
<div>	定义文档中的节
<span>	定义文档中的节
<header>	定义 section 或 page 的页眉
<footer>	定义 section 或 page 的页脚
<section>	定义 section
<article>	定义文章
<aside>	定义页面内容之外的内容
<details>	定义元素的细节
<dialog>	定义对话框或窗口
<summary>	为 <details> 元素定义可见的标题

（11）元信息相关的元素，表4.11列出了一些常用的元信息相关的元素。

表4.11

标签	描述
<head>	定义关于文档的信息
<meta>	定义关于 HTML 文档的元信息
<base>	定义页面中所有链接的默认地址或默认目标
<basefont>	定义页面中文本的默认字体、颜色或尺寸（不赞成使用）

（12）编程相关的元素，表4.12列出了一些常用的编程相关的元素。

表4.12

标签	描述
<script>	定义客户端脚本
<noscript>	定义针对不支持客户端脚本的用户的替代内容
<applet>	定义嵌入的 applet（不赞成使用）
<embed>	为外部应用程序（非 HTML）定义容器
<object>	定义嵌入的对象
<param>	定义对象的参数



4.2 添加文本

文本是HTML页面中最重要的信息之一，它被广泛应用于标题和段落，下面我们就来看一看如何在HTML文档中添加标题和段落，以及对文本进行格式化操作。

4.2.1 标题

在HTML中，标题用<h1>到<h6>的标签进行定义，其中<h1>定义最大的标题，<h6>定义最小的标题。浏览器会自动地在标题的前后添加空行，这样每一个标题就只会显示在一行中。

使用不同的标题，标题中内容是否为粗体和字号都不相同，例如下面这段代码：

```
<h1>1号标题</h1>
<h2>2号标题</h2>
<h3>3号标题</h3>
<h4>4号标题</h4>
<h5>5号标题</h5>
<h6>6号标题</h6>
```

使用h1元素的标题字号最大，到h6元素标题字号依次减小，所有字号均为粗体，每一个标题单独显示一行，即使所有的代码都写在一行，每个标题的前后仍然会添加一个额外的空行。这段代码运行后的效果如图4.1所示。

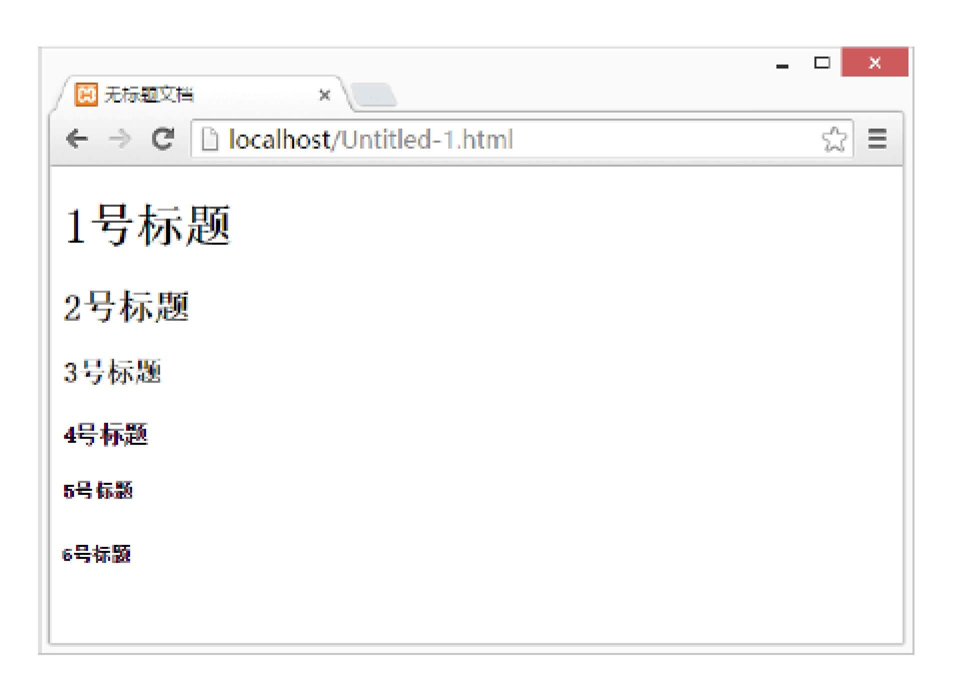


图4.1

在HTML文档中，使用标题来构建文档的索引，不仅可以使用户非常方便地浏览网页的结构，而且还会为搜索引擎提供更好的支持，有利于网页的收录。

4.2.2 段落

在HTML文档中，用<p>标签定义段落，<p>元素开始标签和结束标签中的内容就是段落的内容。例如下面这段代码：

```
<p>HTML段落内容</p>
<p>第一个段落内容</p>
<p>第二个段落内容</p>
```

这段代码中有3个段落，每个段落都由一对<p>标签组成，运行这段代码后，效果如图4.2所示。

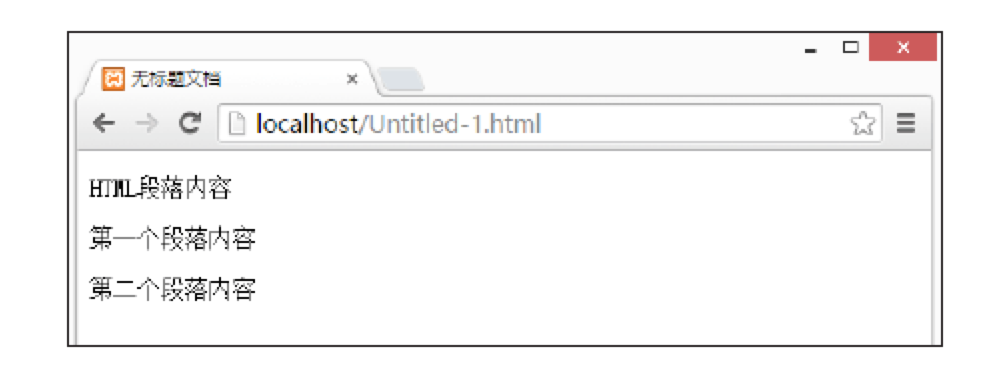


图4.2

浏览器为每个段落的前后添加空行，所以段落与段落之间有一定的距离，这样文章的段落才能更加清晰。这里需要注意的是，<p>元素和
元素都会有一个换行的效果，但是不建议用<p>元素替代
元素的换行效果。

4.2.3　文本的格式化

在HTML文档中，我们经常需要使用一些特殊的格式化文字来展示某些文本，例如斜体、粗体、上标、下标等。为了满足这类需求，HTML提供了一套文本格式化标签，如表4.13所示。

表4.13

标签	描述
<b>	定义粗体文字
<big>	定义大号字
<em>	定义着重文字
<i>	定义斜体字
<small>	定义小号字
<strong>	定义加重语气
<sub>	定义下标字
<sup>	定义上标字
<ins>	定义插入字
<del>	定义删除字

例如下面这段代码，使用各种格式化标签对显示的文本进行格式化操作。运行这段代码后的效果如图4.3所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>文本格式化</title>
</head>
<body>
<p>这是普通文本<br/><b>这是粗体文本</b><br/>
<big>大一号<br/><big>再大一号</big></big><br/>
<em>这是着重文字</em><br/>
<i>这是斜体字</i><br/>
<small>这是小号字</small><br/>
```

```
<strong>这是加重语气文字</strong><br/>
X<sub>2</sub>X<sup>2</sup><br/>
<ins>设置插入文字</ins><br/>
<del>这是删除文字</del><br/>
</body>
</html>
```

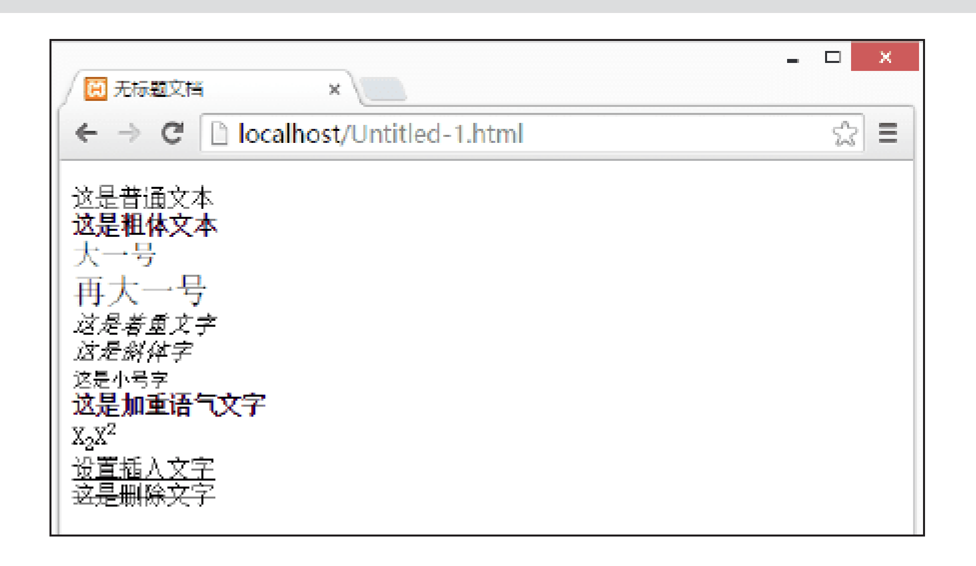


图4.3



4.3 插入图像

图像是HTML网页中另一个非常重要的信息承载对象。无论是背景图像，还是网站logo，甚至是网页中的各种图标，都需要通过插入的方式才能在网页中显示，下面我们来详细介绍如何在网页中插入图像。

4.3.1 在网页中插入图像

使用Dreamweaver在网页中插入图像的方法非常简单，可以参照以下步骤在网页中插入一幅图像。

01 新建一个HTML网页，切换到“设计”视图。

02 选择“插入”→“图像”→“图像”命令，如图4.4所示。

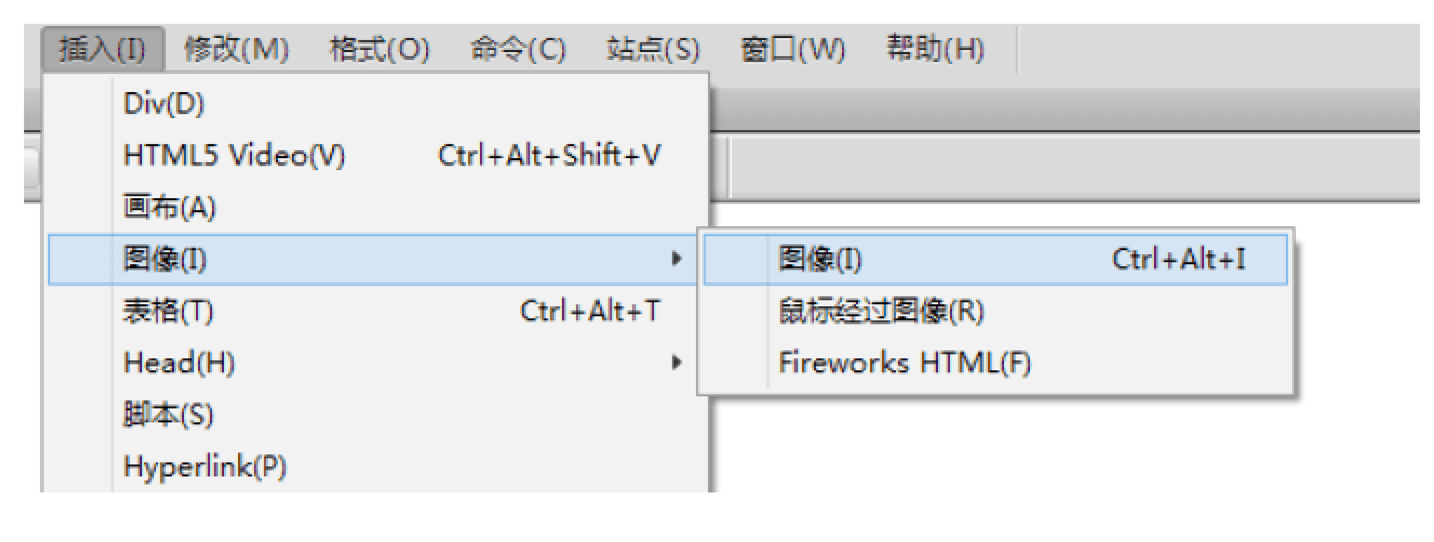


图4.4

（3）在打开的“选择图像源文件”对话框中选择要插入的图像，如图4.5所示。

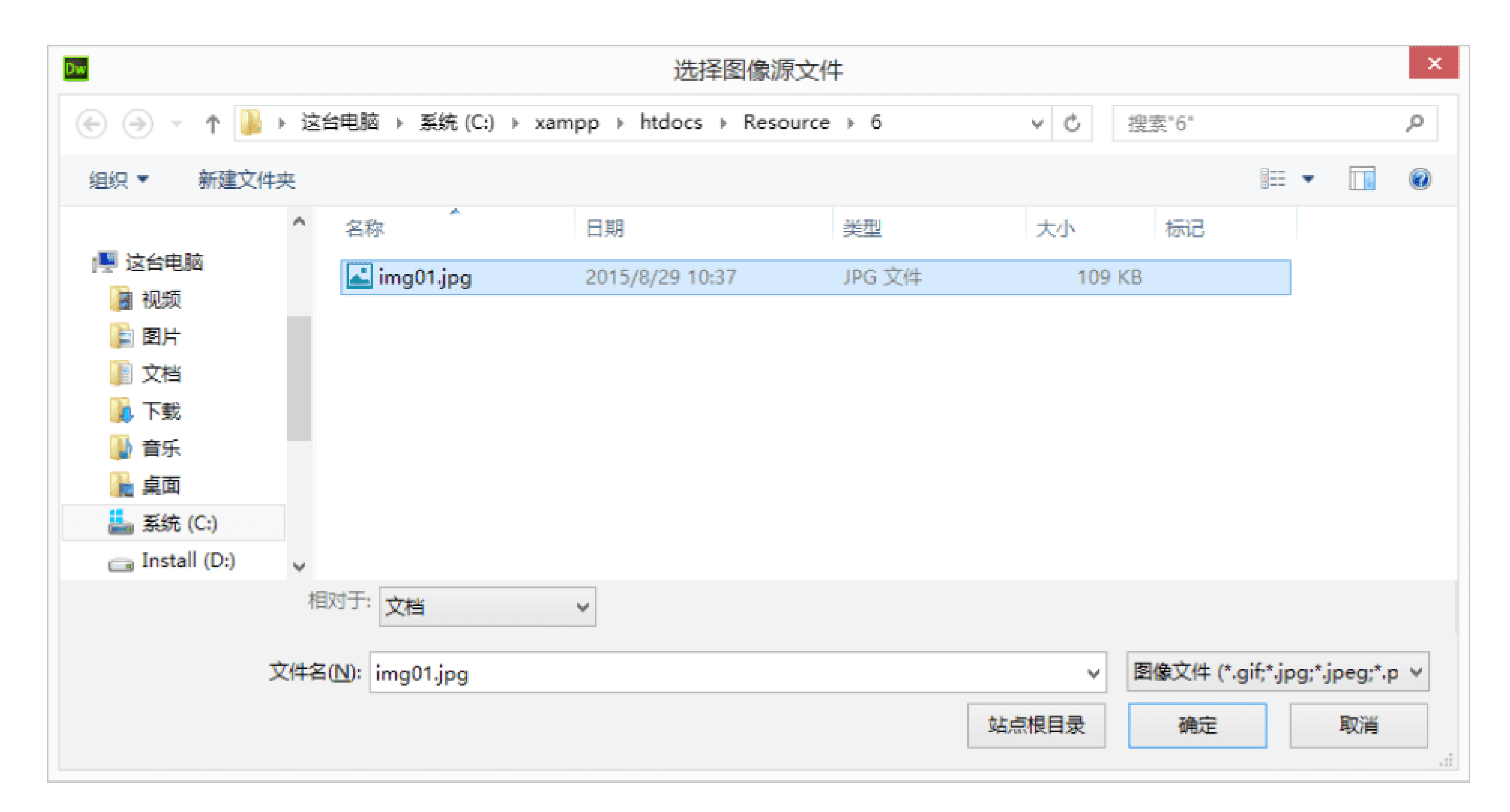


图4.5

（4）单击“确定”按钮后，选中的图像就插入到了网页中，效果如图4.6所示。

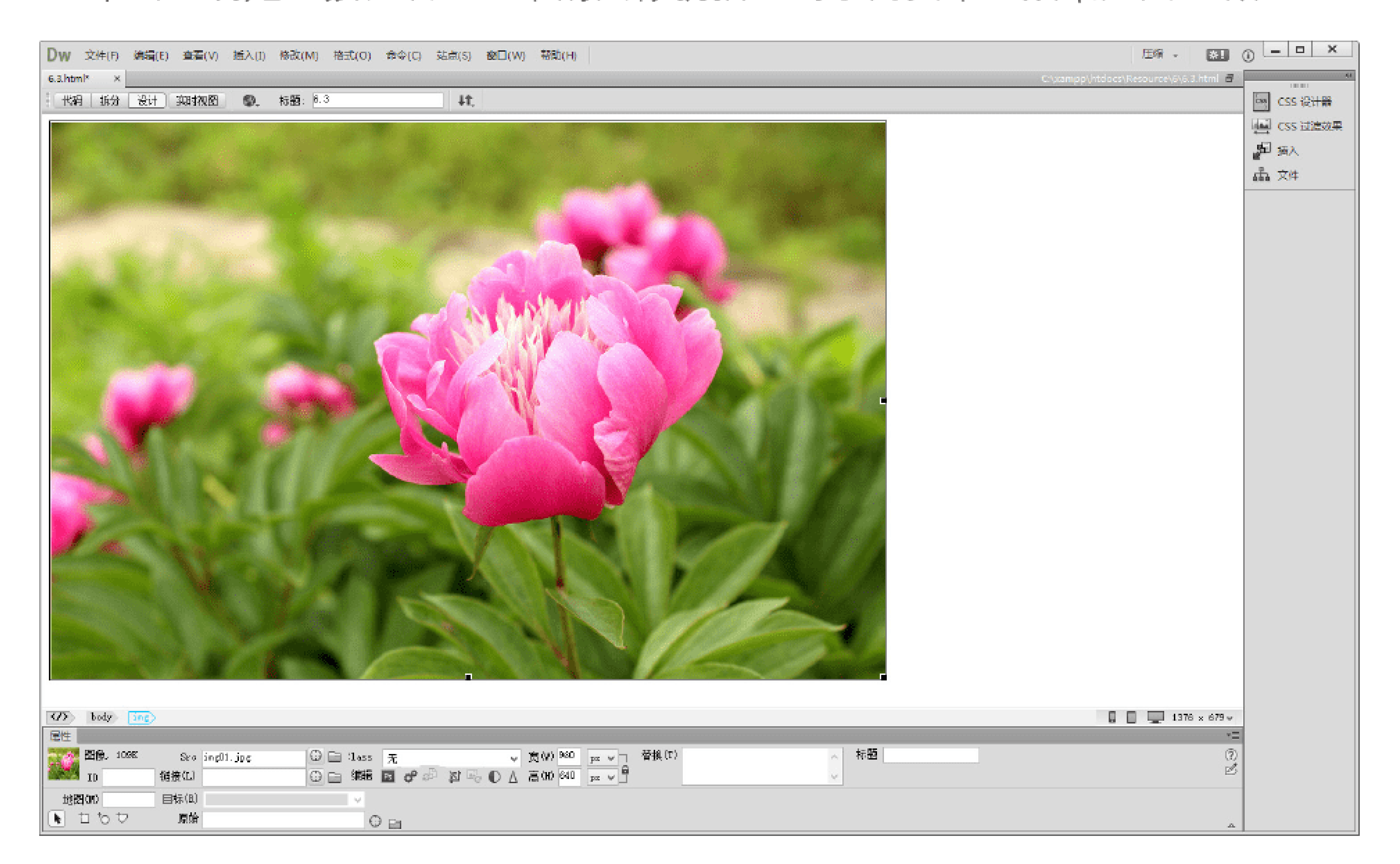


图4.6

4.3.2 图像标签（<img>）和源属性（src）

使用Dreamweaver在HTML页面中插入图像后，我们再切换到“代码”视图，此时可以在<body>标签中看到下面新增加的一行代码：

```
<img src="img01.jpg" width="960" height="640" alt=""/>
```

这行代码由一个<img>标签组成，在HTML中，<img>标签主要用于在网页中插入一个图

像，标签中的src属性指定要插入图像的位置。在上例中，因为图像和HTML网页在同一个文件夹中，所以可以直接写图像的名称，如果图像和HTML网页不在同一个文件夹中，则需要使用绝对路径或相对路径来指定图像的位置。

如图4.7所示，将图像img01.jpg移动到img文件夹中，这样HTML页面与图像所在路径分属不同的目录级别，如果要在HTML页面中正确显示图像，则需要使用如下所示代码：

```
<img src="img/img01.jpg" width="960" height="640" alt=""/>
```

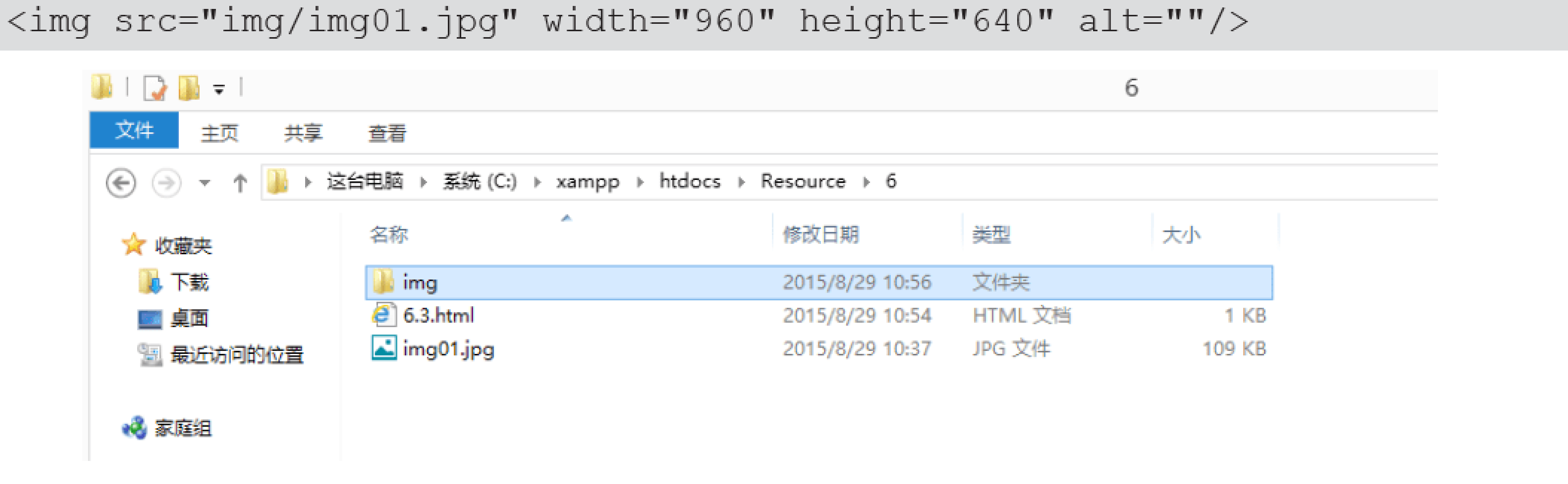


图4.7

举一反三，如果HTML页面与图像分属于不同的磁盘下，在制定图像路径的时候，则需要使用图形的绝对路径。例如，图像img01.jpg位于D盘下的img文件夹中，可以使用如下所示代码，在HTML页面中插入图像。

```
<img src="D:/img/img01.jpg" width="960" height="640" alt=""/>
```

另一种情况，如果要插入的图像并非存储在本地，而是来源于网络，那么该图像就会有一个网络地址，在指定图形路径时，必须使用正确的网络地址才能正确显示图像，例如下面的代码：

```
<img src="http://cdn.duitang.com/uploads/item/201406/18/20140618192710_jiYis.jpeg" width="960" height="640" alt=""/>
```

4.3.3 alt属性

<img>标签中有一个alt属性，该属性用于指定当图像无法显示时的替代文本。以下4种情况下图像将无法显示。

（1）网速太慢。
（2）src属性错误。
（3）浏览器禁用图像。
（4）用户使用的是屏幕阅读器。

当图像无法显示时，网页中图像所在位置会根据不同的浏览器显示不同的效果，如图4.8是谷歌浏览器中图像无法显示的效果，图4.9是IE浏览器中图像无法显示的效果。

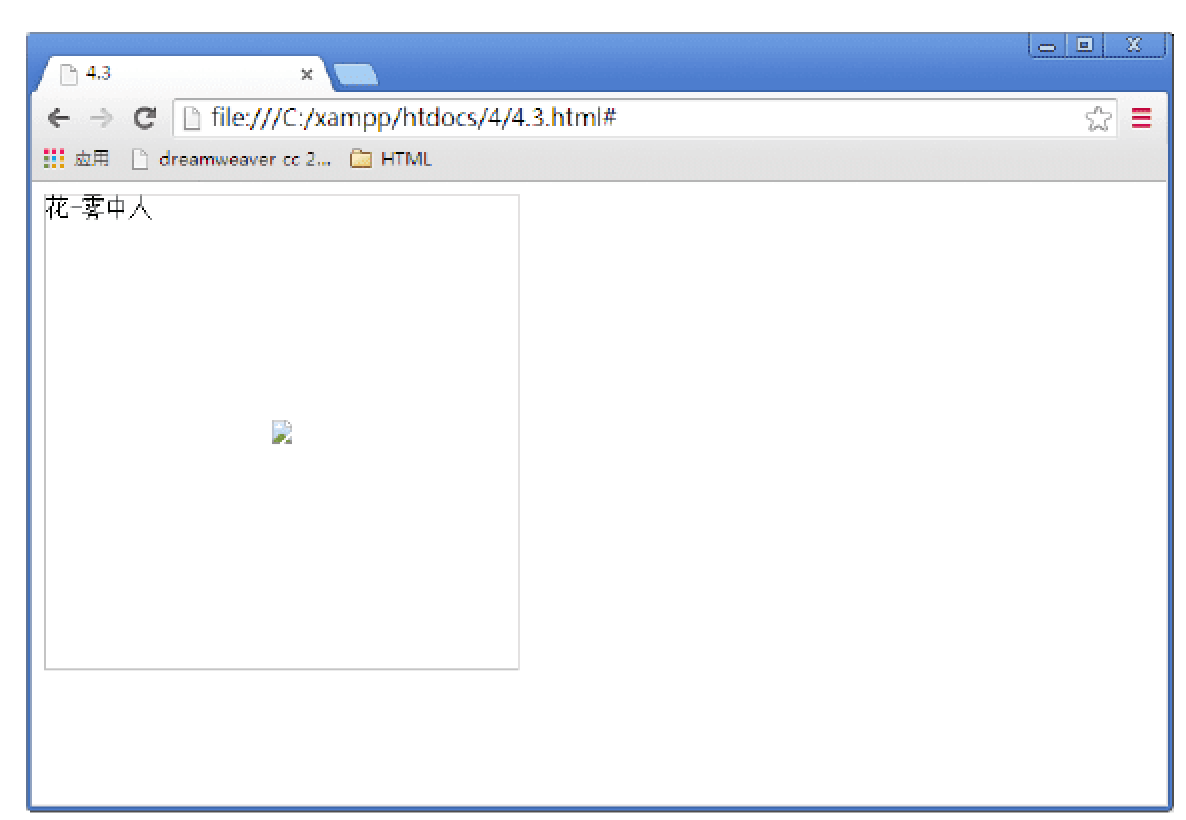


图4.8

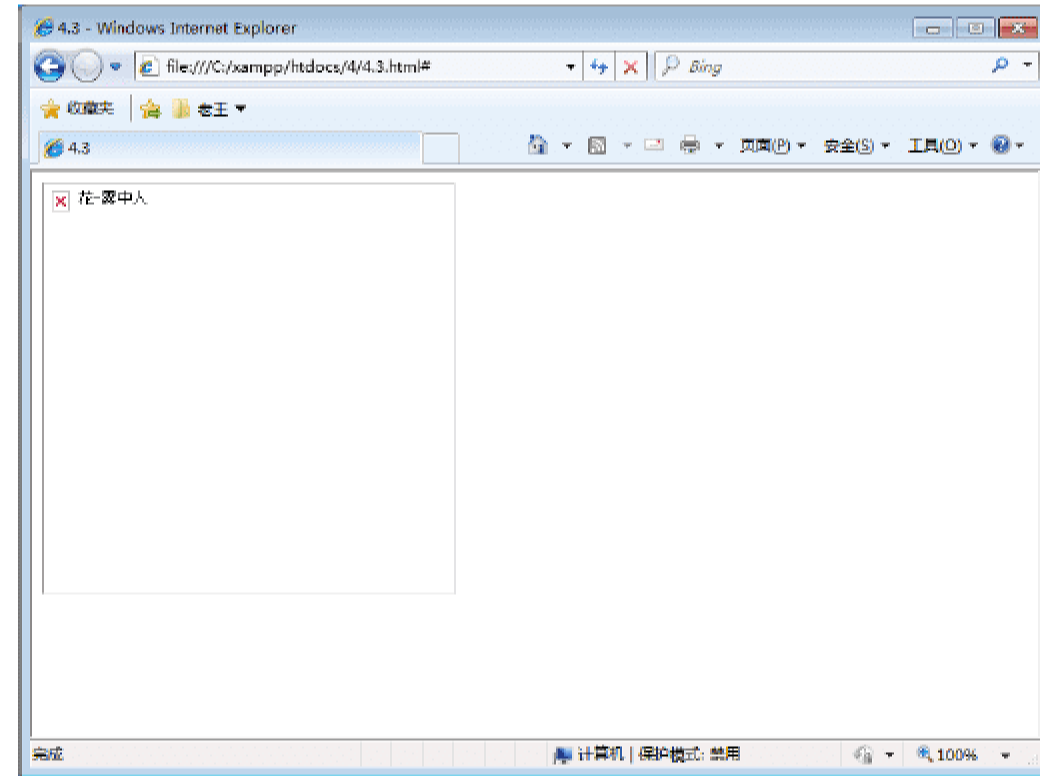


图4.9

以上两幅图中，HTML页面显示的文字就是alt中指定的内容，详细代码如下所示：

```
<img src="img/img02.jpg" width="320" height="320" alt="花-雾中人"/>
```

在这段代码中，需要将图像的名称修改成“img002.jpg”。因为在img文件夹中无法找到“img/img02.jpg”对应的图像文件，所以src指定的图像位置无效。HTML页面中的图像无法正常显示，就会在图像显示的位置用alt属性中指定的内容替代图像。

需要注意的是，<img>元素的alt属性和title属性都可以指定文字内容，alt属性用于指定当图像无法显示时的替代文本，而title属性用于显示一段描述性文字，当鼠标移动到图像所在位置时，在鼠标位置附近会显示这段描述性文字，如图4.10所示。

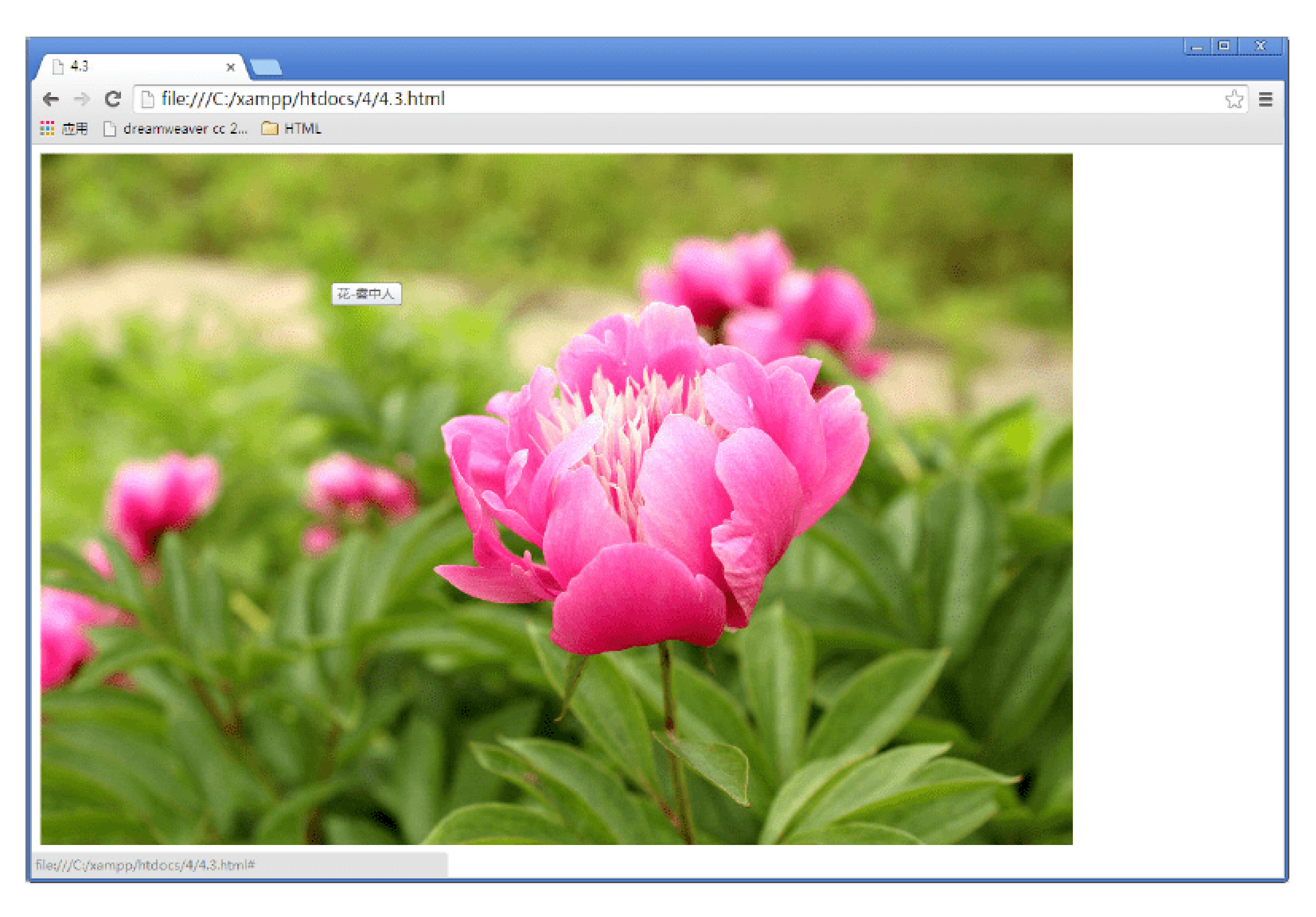


图4.10

4.3.4 从不同的位置插入图像

细心的同学也许已经发现这样一个问题，我们在Dreamweaver中插入的图像始终显示在

HTML页面中的左上角。如果想让插入的图像显示在HTML页面的中间位置，应该如何操作呢？

让我们返回到Dreamweaver操作界面，切换到设计视图。单击选中插入的图像，然后选择“格式”→“对齐”→“居中对齐”命令，如图4.11所示，这样我们在HTML页面中插入的图像就调整到了中间位置，效果如图4.12所示。

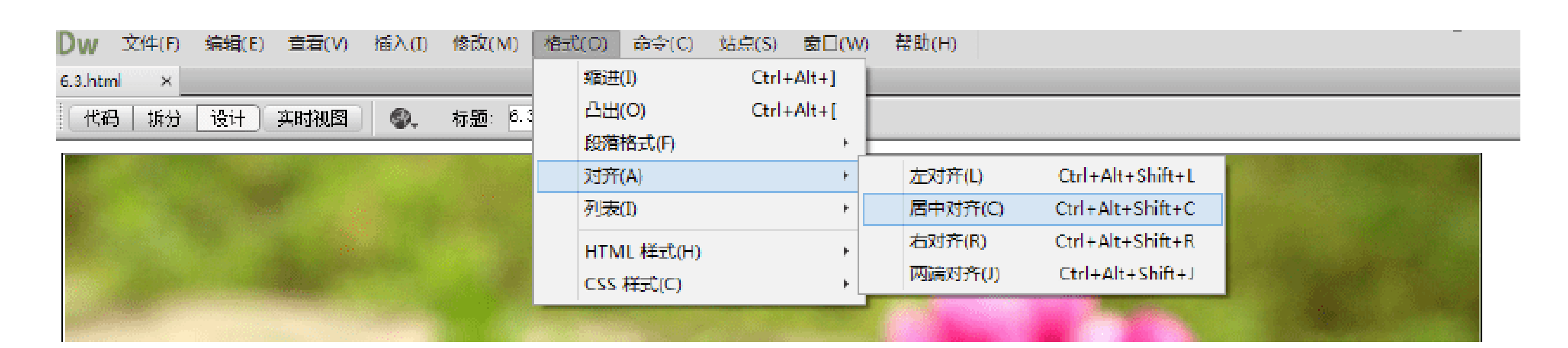


图4.11

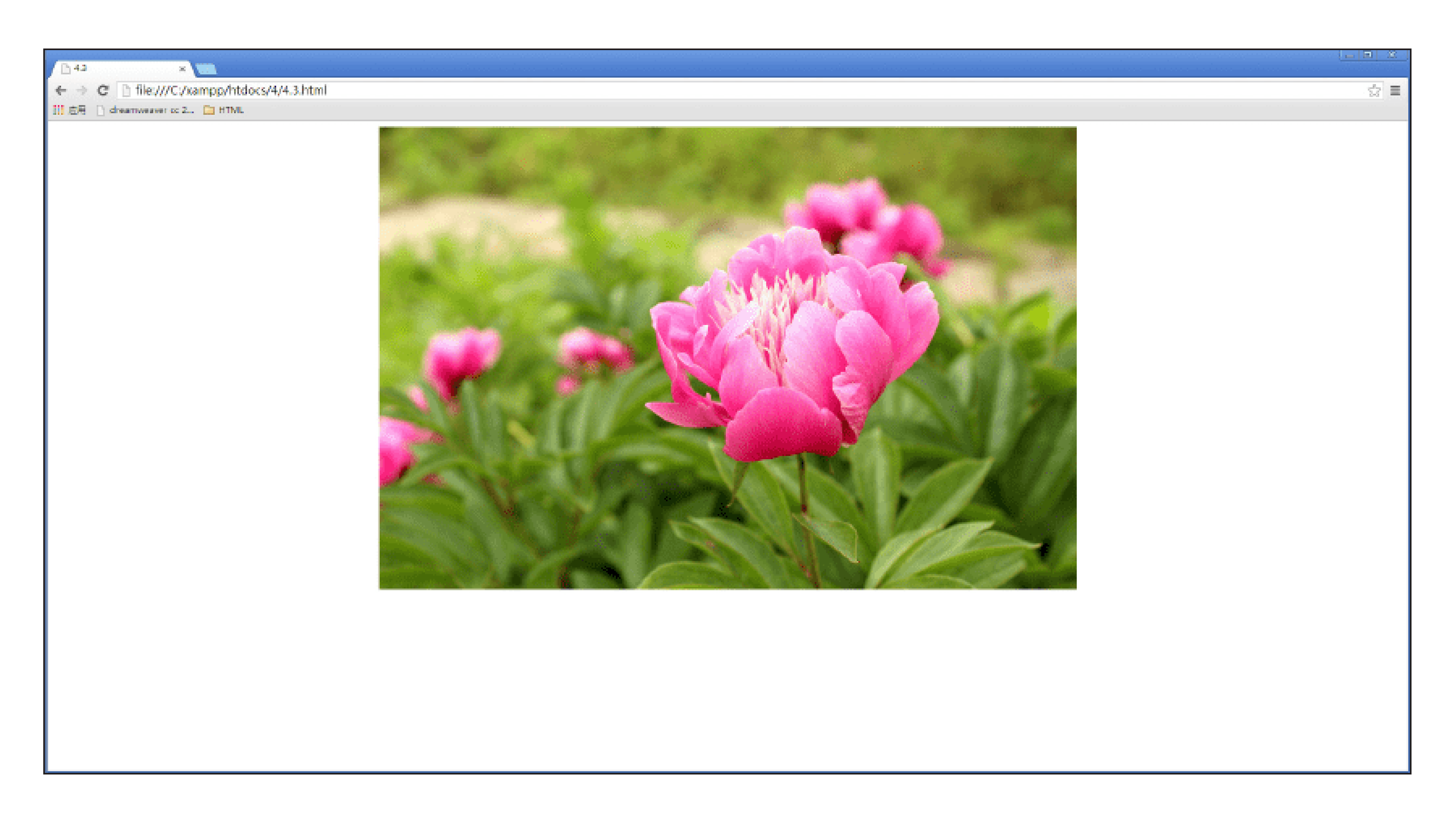


图4.12

返回到代码视图，我们发现在<img>元素的外边新添加了一个<div>元素，并且设置了该元素的align属性为center。也就是说，此时<img>元素作为<div>元素的内嵌元素，并且居中显示，这样就可以达到图像居中显示的效果。相关代码如下所示：

```
<div align="center"><img src="img/img01.jpg" width="960" height="640" alt="花-雾中人" title="这是什么花呢？"/></div>
```

以上设置图像位置的方法比较局限，如果要更加灵活的设置图像的位置，可以使用下面的这种设置，效果如图4.13所示。

```
<img src="img/img01.jpg" width="960" height="640" alt="花-雾中人" title="这是什么花呢？" style=" position:absolute; left:100px; top:50px;"/>
```

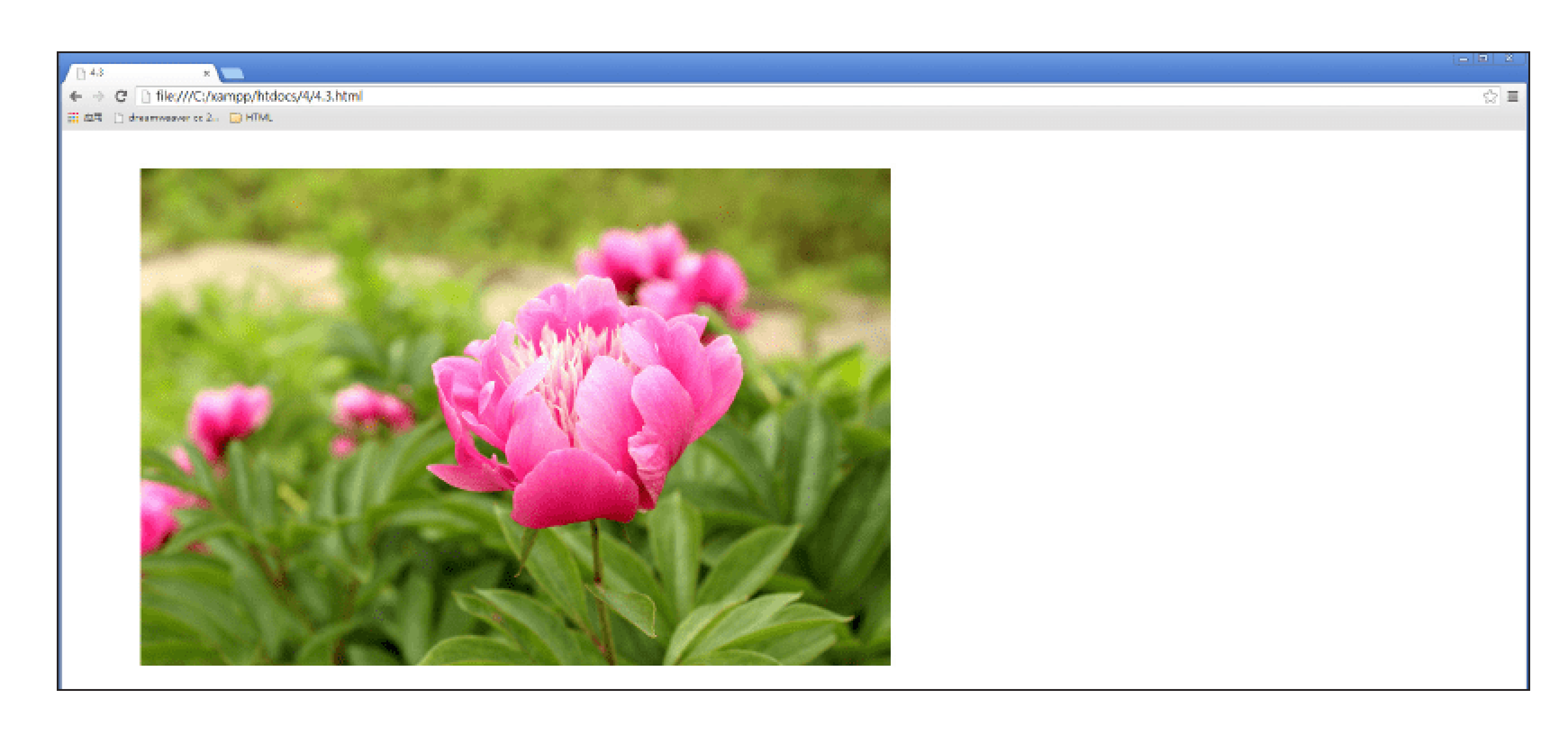


图4.13

在这段代码中，我们添加了style属性，并设置position为absolute，表示当前图像以绝对位置显示，然后设置left属性为100px，top属性为50px，表示当前图像的左上角位置坐标为距离左边界100个像素，距离上边界50个像素，这样就可以灵活控制图像显示的位置了。

4.3.5 定义图像的高度和宽度

在Dreamweaver中插入图像后，默认以图像原始尺寸插入图像，并且在代码中用width表示图像的宽度，用height表示图像的高度，图像尺寸单位为像素（px）。我们还可以根据需要，手动调整图像的高度和宽度。例如，上例中插入的图像宽度为960，高度为640，这个图像在网页中显示的效果太大了，我们需要调整它的尺寸宽度为400，高度为200，刷新页面后的效果如图4.14所示。

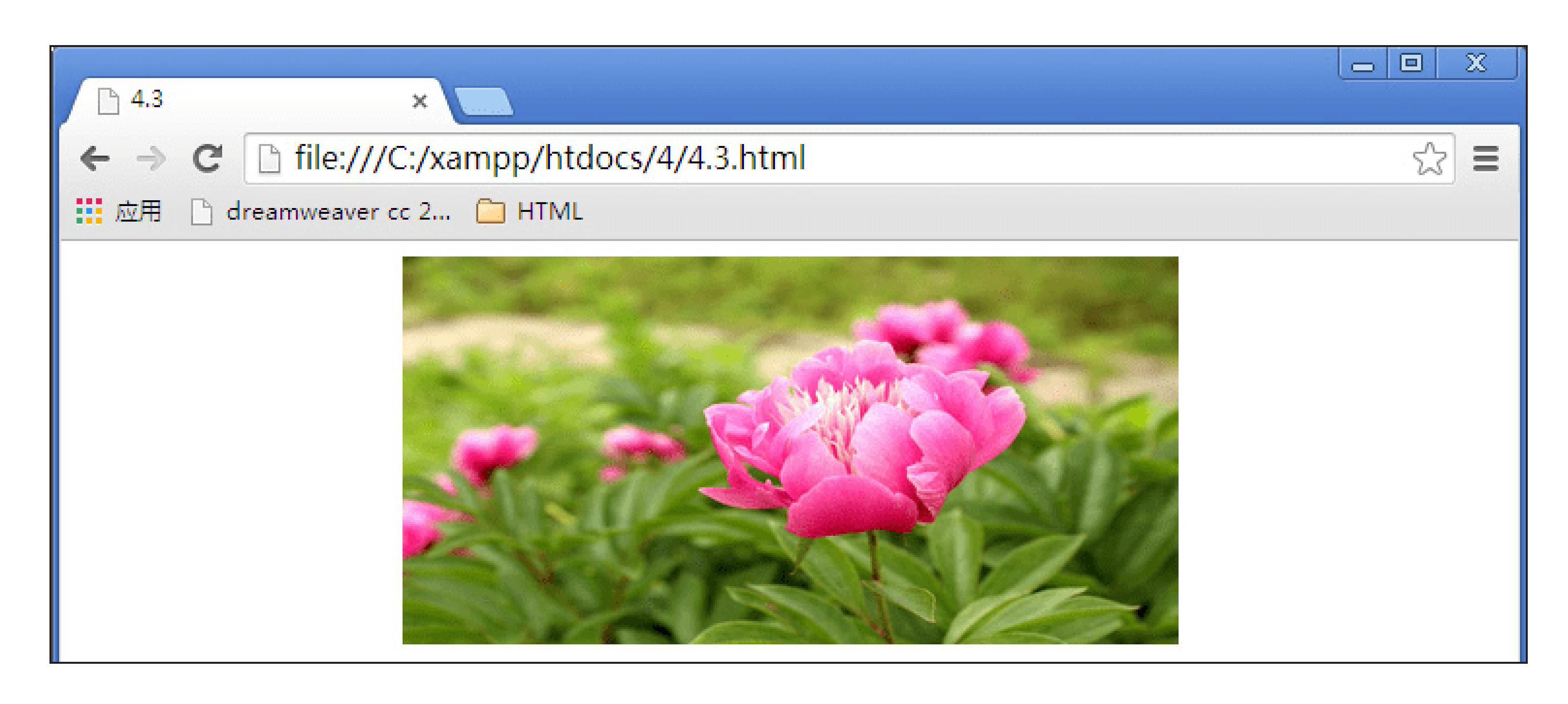


图4.14

调整后的图像效果发生了扭曲变形，这是因为相对图像的原始尺寸，调整后的尺寸失去了宽比约束，我们可以通过等比例缩放图像尺寸避免这种情况。

4.3.6 图像的绕排

在HTML页面中，文字和图像是最重要的信息承载对象。当文字和图像必须同时出现在同一个区域时，就需要使用图像的绕排功能。

<img>元素的align属性用于指定图像及其周围元素的排列方式，align可指定的值为left、

right、top、middle和bottom。left和right会将图像浮动到周围元素的左边或者右边的边界上，top、middle和bottom会将图像与周围文字在垂直方向上对齐。

图片左对齐效果如图4.15所示。



图4.15

图片右对齐效果如图4.16所示。



图4.16

垂直居上对齐效果如4.17所示。

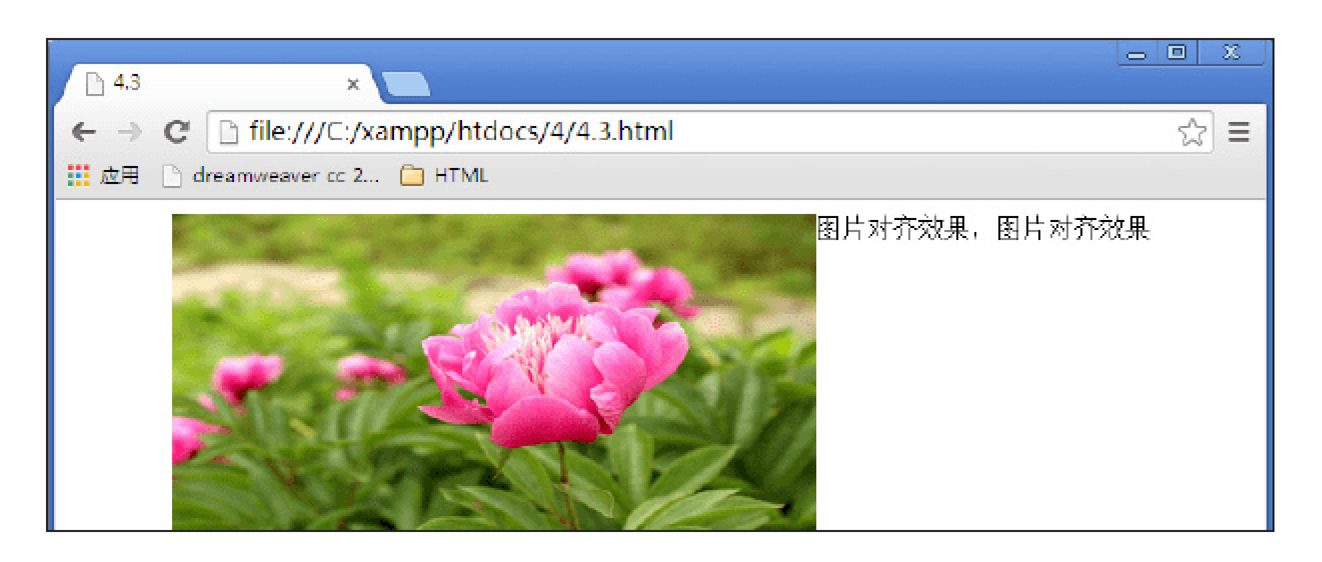


图4.17

垂直居中对齐效果如图4.18所示。

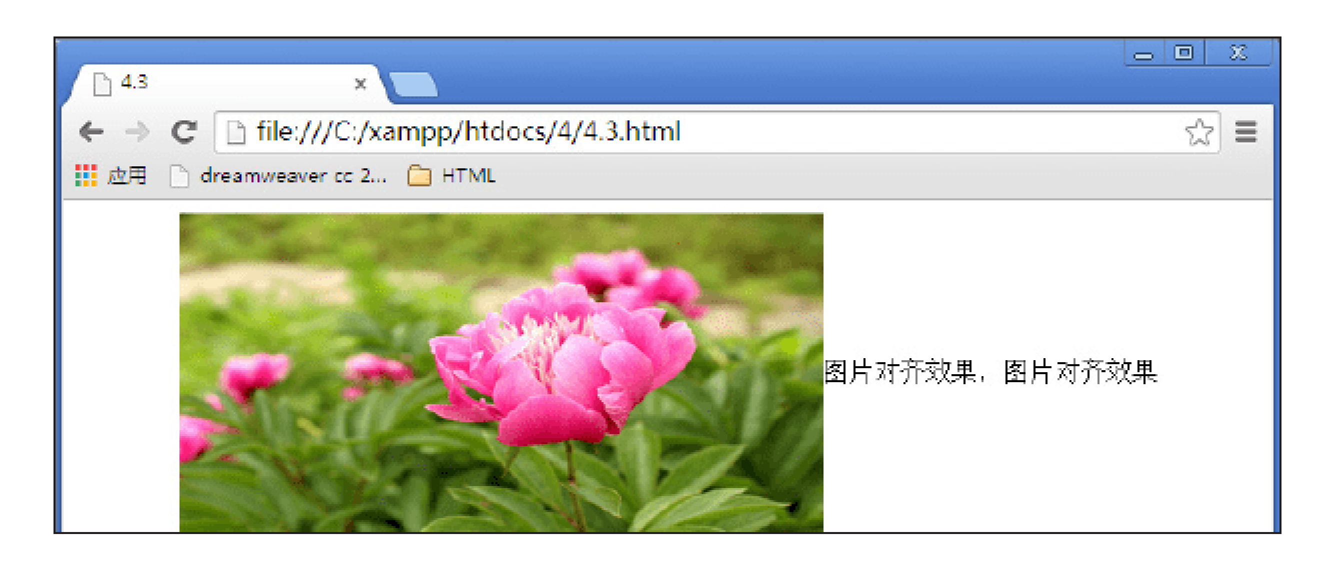


图4.18

垂直居下对齐效果如图4.19所示。

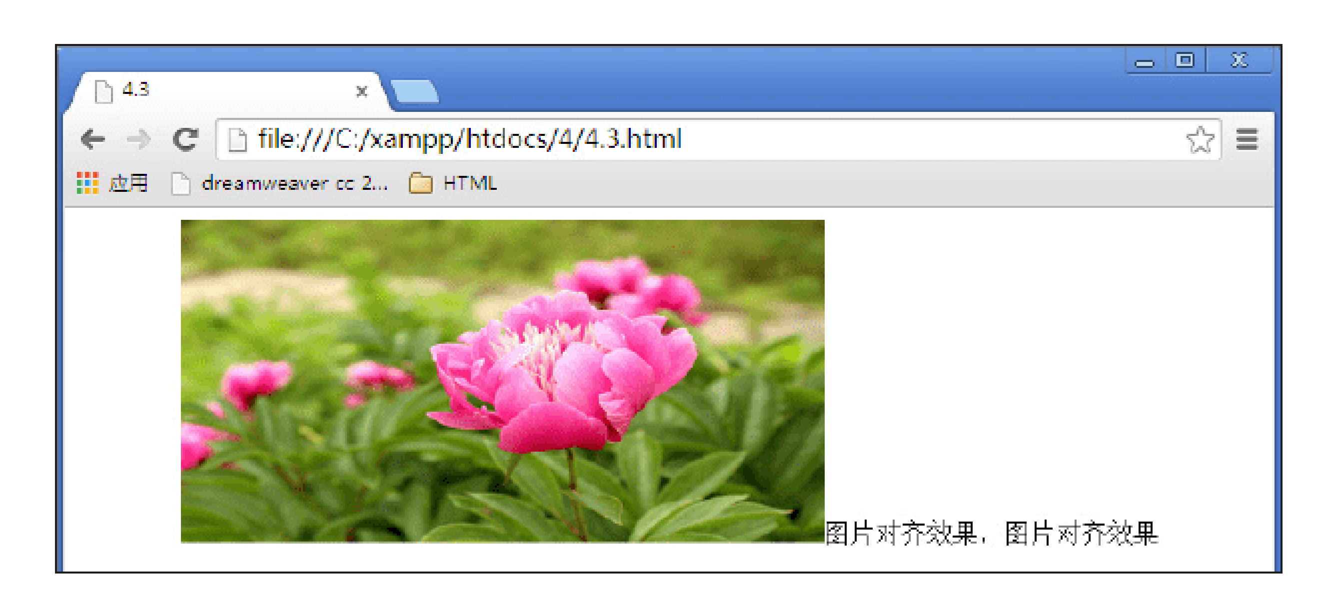


图4.19

4.3.7 创建图像映射

图像映射是指将一个图像划分成多个区域，每一个区域都可以指定一个超链接，点击不同的区域会跳转到不同地址。图像映射最经典的应用就是导航地图，当鼠标移动到地图上一个区域时，该区域的背景颜色呈现为灰色，并显示当前区域的名称，如图4.20所示。

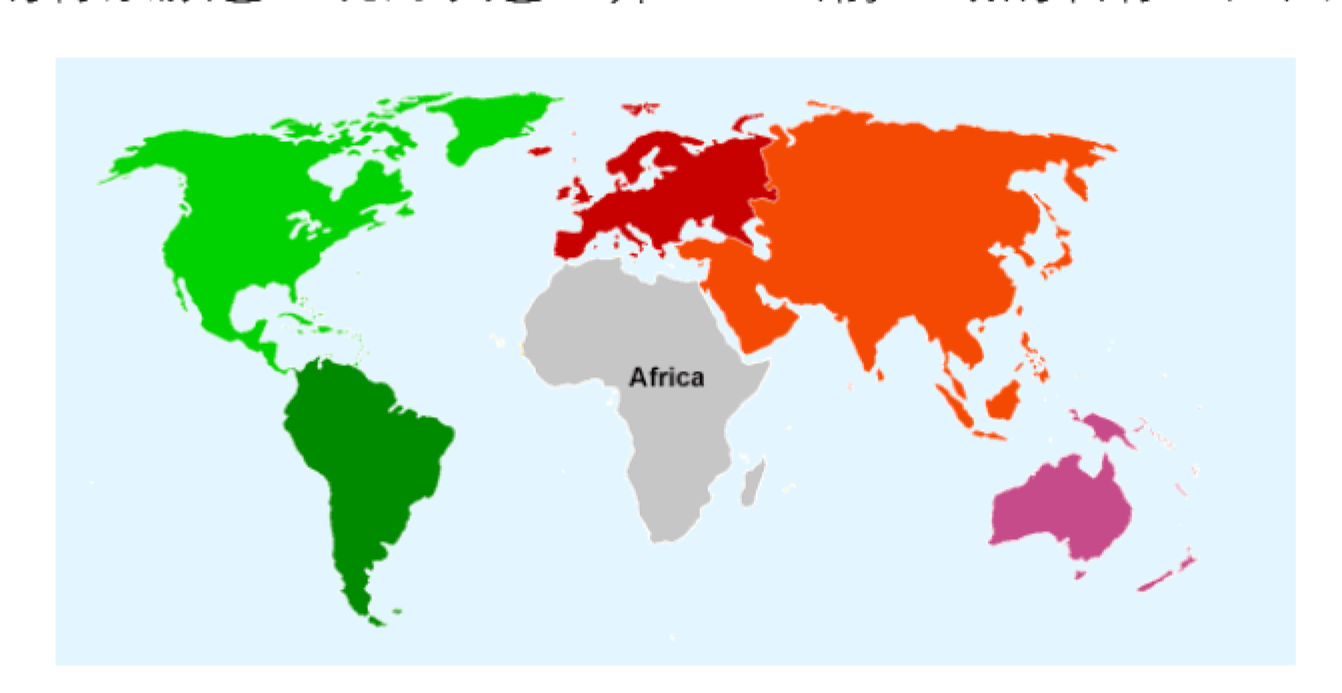


图4.20

在HTML中，创建图像映射的关键在于使用<map>和<area>这两个元素。<map>元素用于指定图像映射的区域，必须为其指定一个name值，该值用于在<img>元素中设置使用哪个图像映射，<area>元素用于在图像地图中划分作用区域。

一个<map>元素中可以有多个<area>元素，每个<area>元素指定一个映射区域。<area>元素中有两个非常重要的属性，shape和coords。shape用于指定映射区域的形状，目前可以设定的值有矩形（rect）、圆形（circle）和多边形（poligon），coords用于指定坐标点，详细使用方法如下：

<area shape="rect" coords="x1, y1,x2,y2" href=url>表示设定热点的形状为矩形，左上角顶点坐标为（x1,y1），右下角顶点坐标为（x2,y2）。

<area shape="circle" coords="x1, y1,r" href=url>表示设定热点的形状为圆形，圆心坐标为（x1,y1），半径为r。

<area shape="poligon" coords="x1, y1,x2,y2 ......" href=url>表示设定热点的形状为多边形，各顶点坐标依次为（x1,y1）、（x2,y2）、（x3,y3）......

在Dreamweaver中，通过可视化操作，可以非常方便地为图像创建映射，详细操作步骤如下：

01 在Dreamweaver中插入一幅图像。

02 切换到设计视图，选中插入的图像，在窗口的下面的属性面板中可以看到用于设置映射的区域，如图4.21所示。

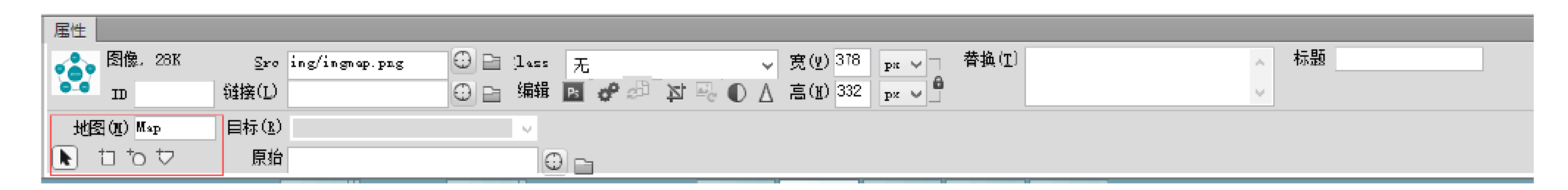


图4.21

03 单击绘制圆形映射的图标，在插入的图形中绘制一个圆形，然后单击图标，移动圆形到合适的位置，效果如图4.22所示。

04 用同样的方法绘制其他映射区域，效果如图4.23所示。



图4.22

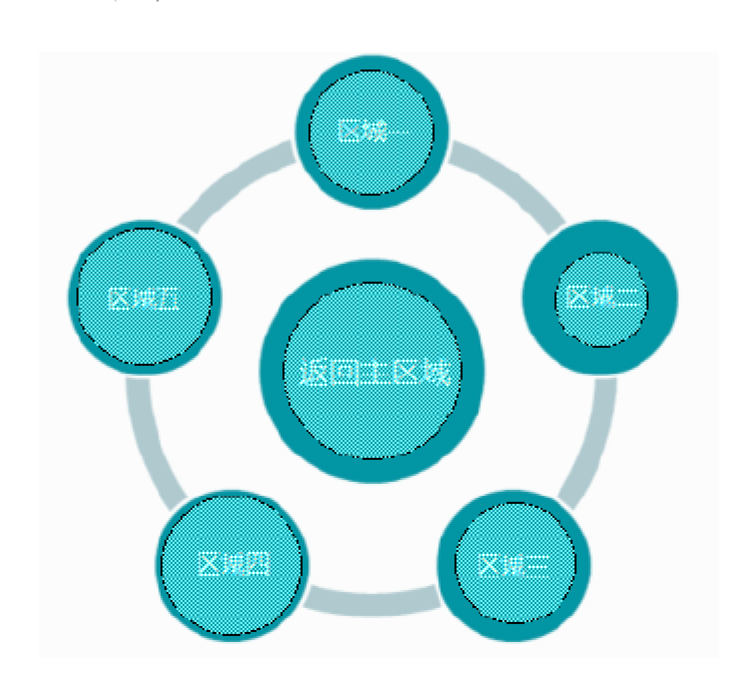


图4.23

（5）用鼠标选中一个绘制的映射区域，在属性面板中的“链接”位置输入一个链接地址，这样当鼠标单击该区域位置时，就会跳转到指定的页面，效果的图4.24所示。

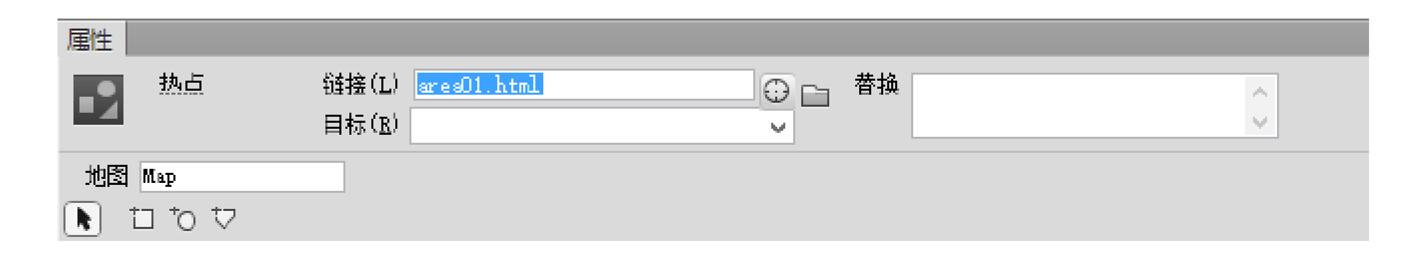


图4.24

（6）用同样的方法为其他映射区域添加链接地址，切换到代码视图，我们可以看到如下所示代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>4.3</title>
</head>
<body>
<img src="img/imgmap.png" alt="" width="378" height="332" usemap="#Map"/>
```

```
<map name="Map">
  <area shape="circle" coords="184,49,33" href="area01.html">
  <area shape="circle" coords="305,137,25" href="area02.html">
  <area shape="circle" coords="185,174,47" href="area00.html">
  <area shape="circle" coords="260,275,32" href="area03.html">
  <area shape="circle" coords="111,276,37" href="area04.html">
  <area shape="circle" coords="65,135,36" href="area05.html">
</map>
</map>
</body>
</html>
```

不难看出，使用Dreamweaver创建的图像映射，其实质还是以代码的形式来创建的。在实际的使用过程中，为了提高工作效率和准确性，使用Dreamweaver比直接使用代码效率更高。



4.4 使用超链接

超链接是网站中使用非常频繁的一种HTML元素，网站中的多个网页就是通过超链接实现了页面间的相互跳转。下面将详细介绍HTML中超链接的使用方法。

4.4.1 链接语法

超链接用<a>标签来表示，其基本语法如下所示：

```
<a href="url">超链接</a>
```

其中href属性用于指定链接目标，链接目标可以是另一个文档的位置，也可以是当前文档的书签，也称为锚。开始标签和结束标签之间的部分是超文本链接的内容。

在超链接中，name属性规定锚的名称。name属性用于创建HTML内部的书签。书签不会以任何特殊方式显示，它对读者是不可见的。当使用命名锚时，我们可以创建直接跳至页面中某个节的链接，这样就可以非常方便地为页面创建目录。

4.4.2 target属性

<a>元素的target属性用于指定在何处打开链接文档。如果在一个<a>标签内包含一个target属性，浏览器将会载入和显示用这个标签的href属性命名的、名称与这个目标吻合的框架或者窗口中的文档。如果这个指定名称或id的框架或者窗口不存在，浏览器将打开一个新的窗口，给这个窗口一个指定的标记，然后将新的文档载入这个窗口。从此以后，超链接文档就可以指向这个新的窗口。

target属性有4个保留的目标名称用作特殊的文档重定向操作，这4个值分别如下。

（1）_blank：浏览器总在一个新打开、未命名的窗口中载入目标文档。

（2）_self：这个目标的值对所有没有指定目标的<a>标签是默认的，它使得目标文档载入并显示在相同的框架或者窗口中作为源文档。这个目标是多余且不必要的，除非和文档标题<base>标签中的target属性一起使用。

（3）_parent：这个目标使得文档载入父窗口或者包含来超链接引用的框架的框架集。如果这个引用是在窗口或者顶级框架中，那么它与目标_self等效。

（4）_top：这个目标使得文档载入包含这个超链接的窗口，用_top目标将会清除所有被包含的框架并将文档载入整个浏览器窗口。

需要注意的是，这些值都是以下划线开头的，所以在设置target窗口或目标时，不能以下划线为开头设置任何框架的name或id的值，否则浏览器将忽略这些字符。

4.4.3 id属性

<a>标记的id属性可用于创建一个HTML文档的书签标记。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>4.3</title>
</head>
<body>
<h2 id="sec01">第一节</h2>
这是第一节中的内容。<br>
<a href="#sec01">1</a>-<a href="#sec02">2</a>-<a href="#sec03">3</a>
<br><br><br><br><br><br><br><br><br><br><br><br>
<h2 id="sec02">第二节</h2>
这是第二节中的内容。<br>
<a href="#sec01">1</a>-<a href="#sec02">2</a>-<a href="#sec03">3</a>
<br><br><br><br><br><br><br><br><br><br><br><br>
<h2 id="sec03">第三节</h2>
这是第三节中的内容。<br>
<a href="#sec01">1</a>-<a href="#sec02">2</a>-<a href="#sec03">3</a>
</body>
</html>
```

在这段代码中，一个页面设置了多个标题，每个标题下面都有一段文字，文字的后面是3个超链接，它们的href属性分别指向3个标题的id属性值，这样就在HTML文档中创建了书签标记，点击对应的超链接就可以在页面中跳转到对应的标题。为了让页面中跳转的效果更加明显，添加了很多的
标签。效果如图4.25所示。



图4.25

4.4.4 创建图片链接

以上介绍的都是为文字创建超链接，除了文字以外，还可以为图片创建超链接。创建图片超链接的方法与创建文字超链接的方法大同小异，将<a>元素的内容部分换成<img>标签即可。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>4.3</title>
</head>
<body>
<a href="#"><img src="img/img01.jpg" width="960" height="640" alt=""/></a>
</body>
</html>
```

创建图片链接其实就是在网页中插入一个图像，然后再给这个图像创建一个超链接的过程。

4.4.5 创建电子邮件链接

电子邮件链接是指当用户点击页面中的一个超链接时，将会启动电子邮件客户端程序。例如下面这段代码：

```
<a href="mailto:zhangsan@163.com">跟我联系</a>
```

当用户点击“跟我联系”这个超链接时，将会启动电子邮件客户端程序，新建一封准备发往zhangsan@163.com的电子邮件，此时收件人地址默认为zhangsan@163.com。

这里需要注意的是，只有当用户的电脑安装了电子邮件客户端程序才能使用该功能。



4.5 创建用户信息页面

文本、图像和超链接是HTML页面中使用最多的3种元素，掌握这些元素的使用方法，就可以创建一个有模有样的HTML页面。本节内容主要根据前面所学知识，创建一个用户信息展示的HTML页面，最终效果如图4.26所示。详细操作步骤如下：



图4.26

01 在Dreamweaver中新建一个空白的HTML页面。

02 分析如图4.25所示效果，我们可以用<img>标记插入图片，用于展示用户照片。基本信息的分类标题可以用<h3>标记展示，如“基本信息”“学校信息”和“个人信息”。照片下面与用户互动的文字可以设置为超链接，当击这些超链接的时候，跳转到其他页面。其余的文字信息可以用段落标记<p>或者文本格式化标记。

03 页面基础布局。该页面整体可以分为左右结构两部分，左侧又可以再分为上中下结构3部分，我们分别用div标记表示各个部分。为了让各个部分看的更清晰，我们为每个div标记设置了背景颜色，相关代码如下所示：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>用户信息展示</title>
<style>
body { font:12px "宋体","Arial Narrow",HELVETICA; margin:0;}
#container {margin:0 auto; width:550px; }
#sidebar { float:left; width:225px; height:500px;background-color:#D1A3A4;
}
```

```
    #content { float:right; width:325px; height:500px; background-color:#DBF1F1;}
    </style>
    </head>
    <body>
    <div id="container">
      <div id="sidebar">
        上面用img元素显示图像<br>
        中间用div元素显示多个超链接<br>
        下面用table显示基本信息
      </div>
      <div id="content">
    这里用table显示用户信息
      </div>
    </div>
    </body>
    </html>
```

在浏览器中查看效果，如图4.27所示。

04 用img元素在左侧插入图像，相关代码如下所示：

```
    <img src="img/img02.png">
```

在浏览器中查看效果，如图4.28所示。

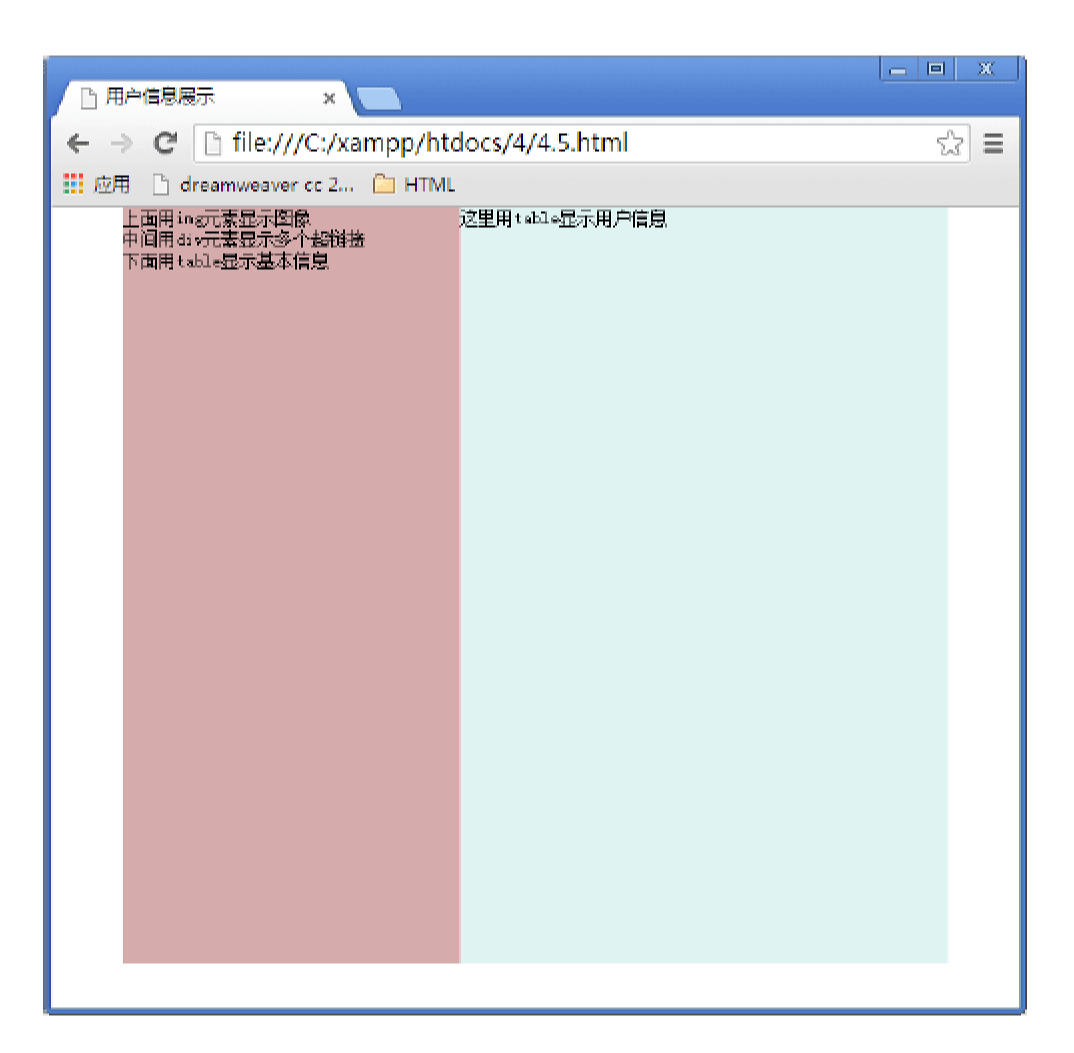


图4.27

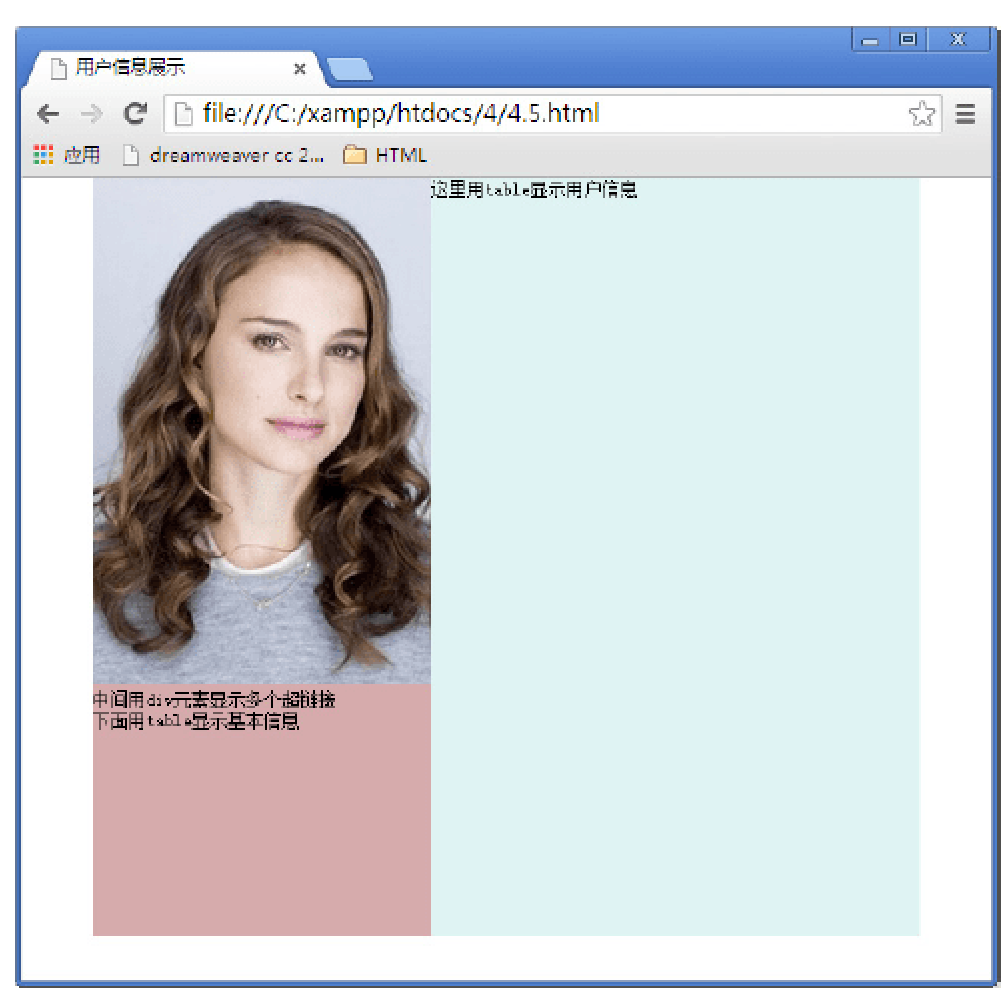


图4.28

05 在img元素下面创建一个div元素，这个div元素内嵌了两个超链接和一个邮件地址，并用b标记加粗这些字体，相关代码如下：

```
    <style>
    #linkitems{float:left; width:225px; margin:10px 0; padding-left:15px; }
    a{text-decoration:none;}
    </style>
    <div id="linkitems">
```

```
        <B><a href="#">向她打招呼</a></B><br>
        <B><a href=" mailto:zhangsan@lycn.com ">发站内信</a></B><br>
        <B><a href="#">送她礼物</a></B><br>
    </div>
```

在浏览器中查看效果，如图4.29所示。

06 在超链接的下面添加一个table标记，用于显示一些基本信息，相关代码如下：

```
    <style>
    table{ margin-left:10px;}
    .tdleft{text-align:left; color:#2B7BB5}
    .tdright{text-align:right; font-weight:bold;}
    </style>
    <table id="tableA">
        <tr><td  class="tdright">所在学校：</td><td  class="tdleft">中国××大学</
td></tr>
        <tr><td class="tdright">生    日：</td><td class="tdleft">1985年3月12日</
td></tr>
        <tr><td class="tdright">家    乡：</td><td class="tdleft">中国西安</td></
tr>
        <tr><td class="tdright">等    级：</td><td class="tdleft">12级</td></tr>
    </table>
```

在浏览器中查看效果，如图4.30所示。

（7）在右侧的div元素中添加一个table元素，使用th标记显示用户信息分类的标题，使用td标记显示用户基本信息，相关代码如下：

```
    <style>
    .thtitle{text-align:left}
    #tableB tr {height:22px;}
    </style>
    <table id="tableB">
        <tr><th colspan="2" class="thtitle"><h3>基本信息</h3></th></tr>
        <tr><td class="tdright">性别：</td><td class="tdleft">女</td></tr>
         <tr><td  class="tdright">生日：</td><td  class="tdleft">1985年3月12日</
td></tr>
        <tr><td class="tdright">家乡：</td><td class="tdleft">中国西安</td></tr>
        <tr><th colspan="2" class="thtitle"><h3>学校信息</h3></th></tr>
        <tr><td class="tdright">大学：</td><td class="tdleft">××大学 ××专业</
td></tr>
        <tr><td class="tdright">高中：</td><td class="tdleft">××中学 - 2003年</
td></tr>
        <tr><td class="tdright">初中：</td><td class="tdleft">××中学 - 2000年</
td></tr>
        <tr><td class="tdright">小学：</td><td class="tdleft">××小学 - 1997年</
td></tr>
        <tr><th colspan="2" class="thtitle"><h3>学校信息</h3></th></tr>
         <tr><td  class="tdright">兴趣爱好：</td><td  class="tdleft">唱歌 跳舞</
td></tr>
        <tr><td class="tdright">喜欢音乐：</td><td class="tdleft">民谣歌曲</td></
```

```
tr>
        <tr><td class="tdright">喜欢电影：</td><td class="tdleft">科幻电影</td></
tr>
        <tr><td class="tdright">玩的游戏：</td><td class="tdleft">策略游戏</td></
tr>
        <tr><td class="tdright">喜欢动漫：</td><td class="tdleft">日本动漫</td></
tr>
        <tr><td class="tdright">喜欢运动：</td><td class="tdleft">跑步</td></tr>
        <tr><td class="tdright">喜欢书籍：</td><td class="tdleft">古典名著</td></
tr>
    </table>
```

在浏览器中查看效果，如图4.31所示。

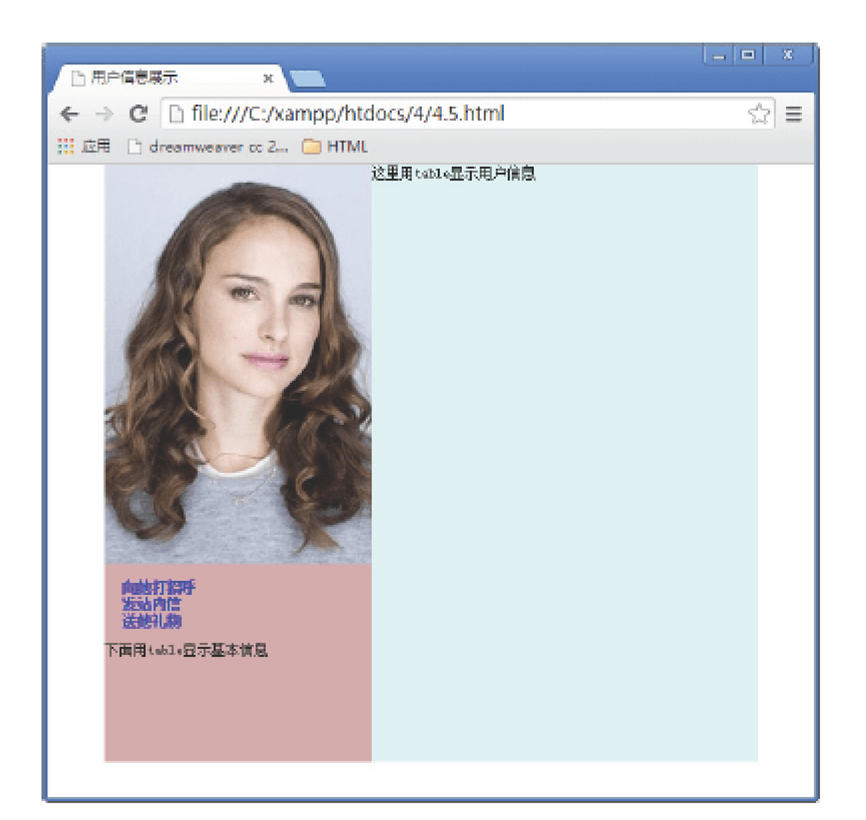


图4.29

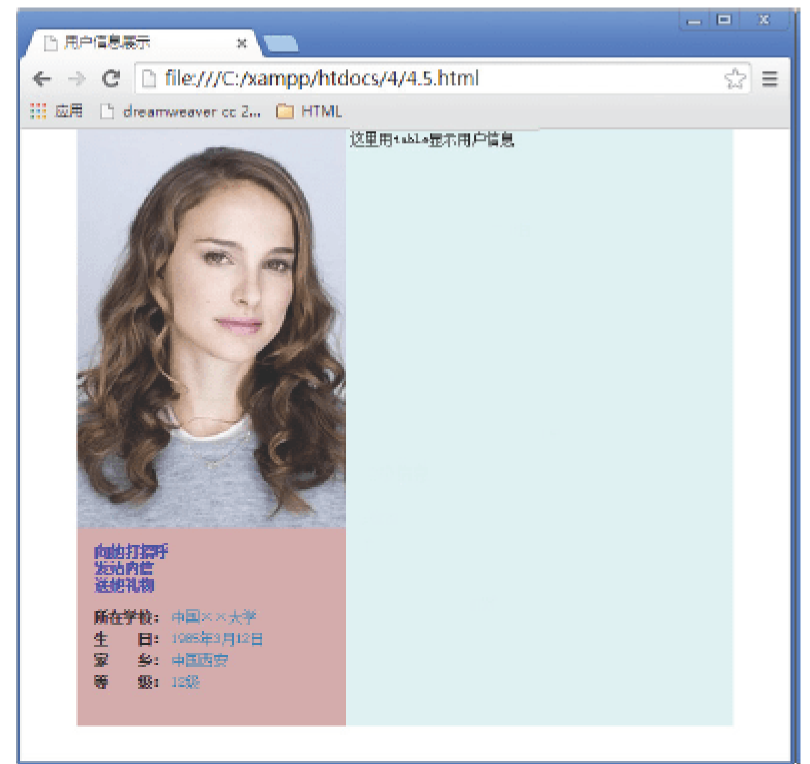


图4.30



图4.31

（8）将左侧背景色修改成与右侧背景色相同的样式，相关代码如下：

```
    #sidebar { float:left; width:225px; height:500px;background-color:#DBF1F1;
}
```

页面最终效果如图4.26所示。

第5章 使用表格与列表组织内容

表格和列表是HTML页面中用于组织页面内容的重要元素，无论是文字、图片或是其他素材，都可以使用表格或者列表进行合理的组织和布局。



5.1 插入表格

在HTML中，表格是用于组织页面内容的重要元素之一。在早期的HTML页面布局中，表格发挥了重要的作用，下面我们就来详细介绍表格的使用方法。

5.1.1 表格的作用

在早期的HTML页面中，表格经常会用于网页布局。打开网页的HTML代码，随处可见表格以及表格中嵌套的表格，使用表格的目的只有一个，那就是布局一个非常漂亮的网页。随着技术的发展和对网页结构的要求更高，现在的表格已经不再承担页面布局的作用，更多是作为承载数据的容器，因为表格排列数据的功能还是非常完美的。

5.1.2 表格的结构

在HTML页面中，表格用<table>标签表示，除此之外，还有很多标签用于表示表格的结构，详细如下所示：

（1）<caption>标签定义表格的标题。
（2）<col>标签为表格的列定义属性。
（3）<colgroup>标签定义表格列的分组。
（4）<thead>标签定义表格的表头。
（5）<th>标签定义表格的表头。
（6）<tr>标签定义表格的行。
（7）<tfoot>标签定义表格的脚注。
（8）<tbody>标签定义表格的主体。
（9）<td>标签定义一个单元格。

以下代码表示了一个表格的基本结构，表格在HTML页面中显示的效果如图5.1所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>5.1</title>
</head>
<body>
<table border="1">
<caption>收支明细</caption>
<colgroup>
 <col />
 <col span="2" />
</colgroup>
<thead>
 <tr>
  <th>日期</th>
  <th>收入</th>
  <th>支出</th>
 </tr>
</thead>
<tfoot>
 <tr>
  <td colspan="3">小计：370</td>
 </tr>
</tfoot>
<tbody>
 <tr>
  <td>2015年7月10日</td>
  <td>100</td>
  <td>30</td>
 </tr>
 <tr>
  <td>2015年7月11日</td>
  <td>200</td>
  <td>150</td>
 </tr>
 <tr>
  <td>2015年7月12日</td>
  <td>300</td>
  <td>50</td>
 </tr>
</tbody>
</table>
</body>
</html>
```

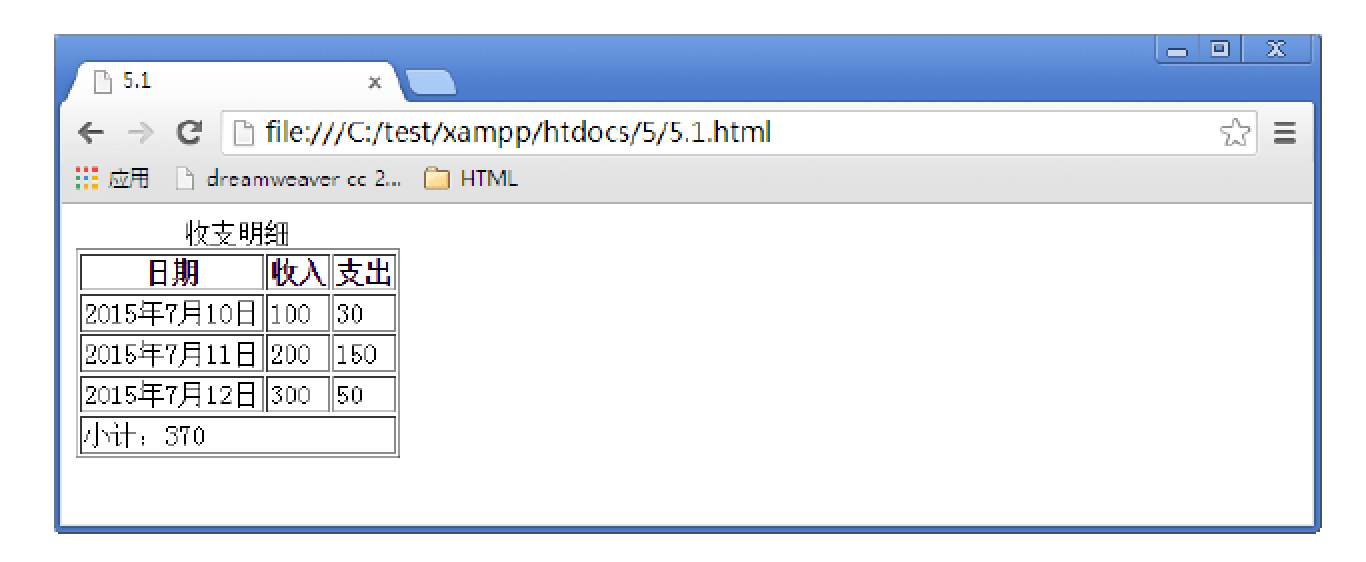


图5.1

5.1.3 在单元格中添加内容

<td>标签用于定义表格中的单元格，<td>标签中的内容也就是单元格中将要显示的内容。如果<td>元素中的内容是文字，那么表格单元格中的内容就会显示文字；如果<td>元素中内容是图片，那么表格单元格中的内容就会显示图片；如果<td>元素中的内容是超链接，那么表格单元格中的内容就会显示超链接，如图5.2所示。

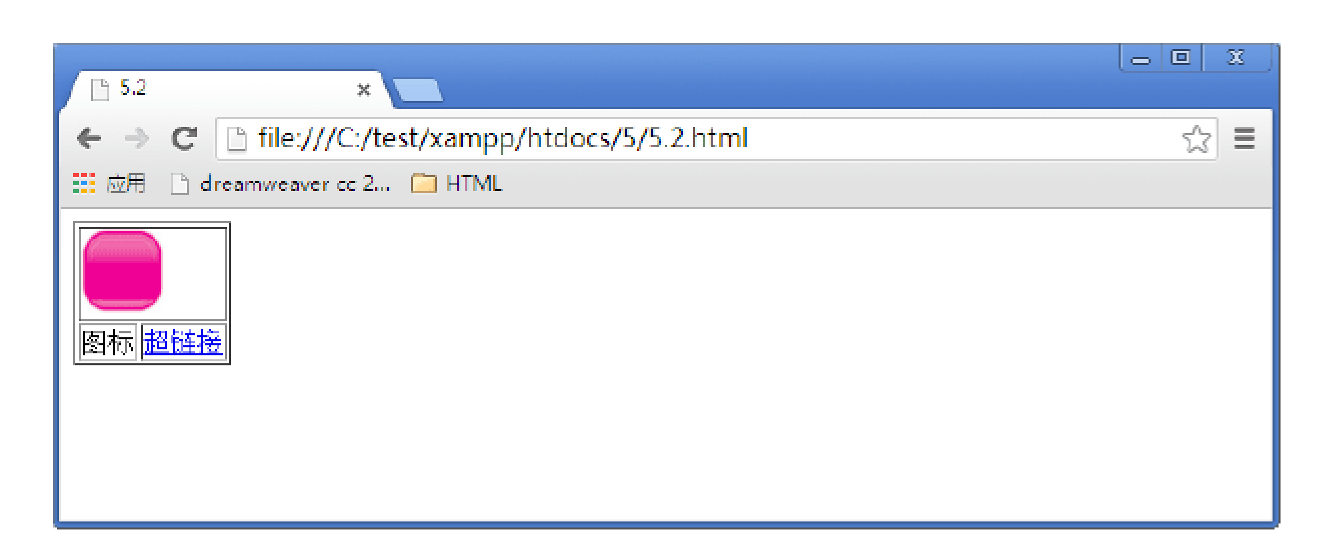


图5.2

表格作为承载其他元素的容器，使页面布局达到整齐划一的效果。如果要在HTML页面中显示表格，则需要为表格设置边框或者背景色。



格式化表格

在HTML页面中，没有经过任何修饰的表格是极其丑陋的。为了让表格与其承载的内容都能以漂亮的面貌呈现出来，就需要对表格进行格式化处理。

5.2.1 id属性

id属性规定了HTML元素的唯一性。在HTML页面中，绝大多数的元素都有id属性。如果为某个元素设置了一个id属性值，那就不能为其他元素设置相同的属性值，否则当使用

JavaScript或者CSS样式的时候，就不能正确获取该元素，无法对其进行其他操作。

5.2.2 class属性

class属性规定了元素的类名。在一个网页中，多个HTML元素可以使用相同的class属性名称。class属性大多数时候用于指向样式表中的类，也可以利用它通过JavaScript来改变带有指定class的HTML元素。

5.2.3 表格的宽度和高度

默认情况下，表格的宽度和高度可以根据单元格中的内容自动调整。我们也可以通过设置表格或者单元格的宽度和高度，手动改变表格的宽度和高度。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>5.3</title>
</head>
<body>
<table border="1" cellspacing="0" width="400px" height="300px">
<tr height="50px">
    <th width="50px">姓名</th>
    <th width="100px">家庭住址</th>
    <th>联系方式</th>
</tr>
<tr>
    <td height="100">张三</td>
    <td>北京市朝阳区</td>
    <td>18644446666</td>
</tr>
<tr>
    <td>李四</td>
    <td>北京市海淀区</td>
    <td>13122225555</td>
    </tr>
</table>
</body>
</html>
```

这段代码运行后的效果如图5.3所示。



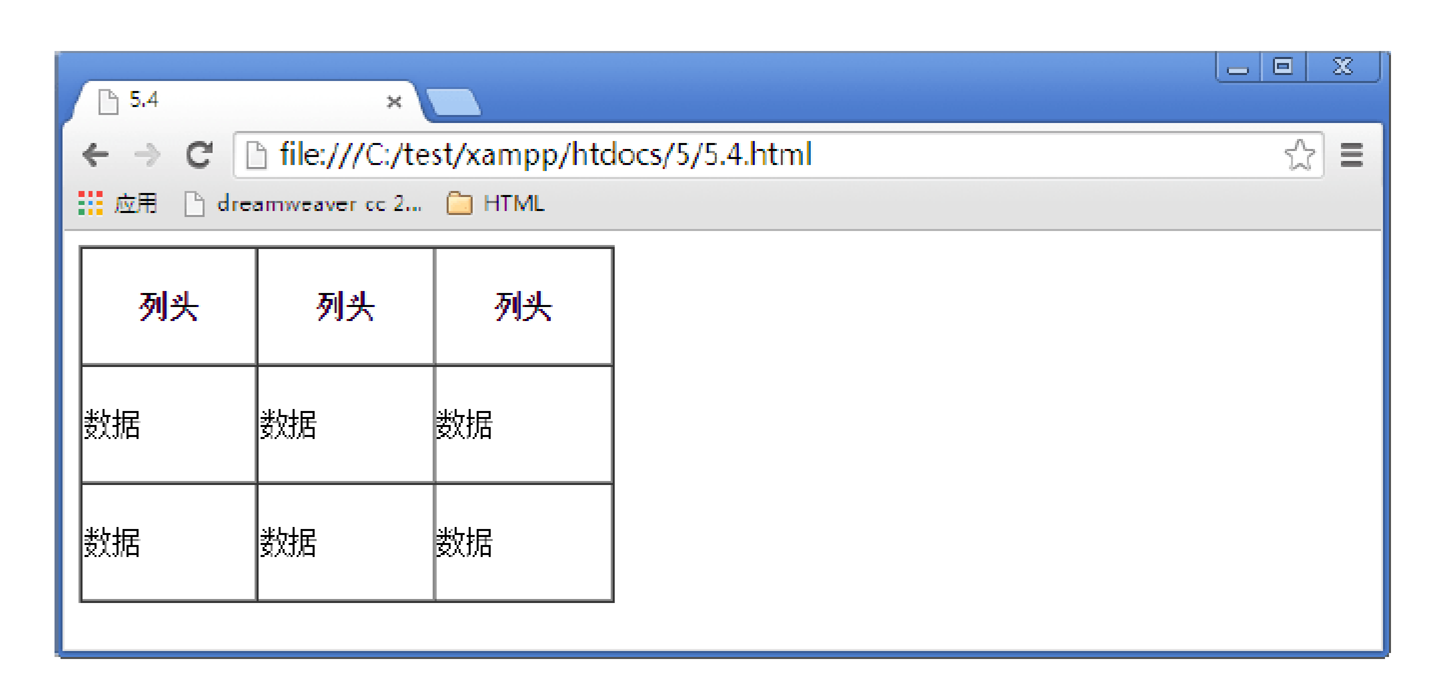
姓名	家庭住址	联系方式
张三	北京市朝阳区	18644446666
李四	北京市海淀区	13122225555

图5.3

从上面的代码中可以看出，<table>元素的宽度和高度可以控制整个表格的尺寸，而数据行的高度可以通过<tr>标签或者<td>标签的height属性进行设置，数据列的宽度可以通过首行单元格的宽度进行设置，如<th>标签或者<td>标签。

5.2.4 表格与单元格的对齐

默认情况下，<td>标签中单元格的内容在水平方向上左对齐，垂直方向上居中对齐，而<th>标签中单元格的内容在水平方向上居中对齐，垂直方向上也是居中对齐，如图5.4所示。



列头	列头	列头
数据	数据	数据
数据	数据	数据

图5.4

在HTML中，align属性控制表格单元格在水平方向上的对齐方式，可以设置的值有以下几种。

（1）left：默认的对齐方式，左对齐内容。
（2）right：右对齐内容。
（3）center：居中对齐内容。
（4）justify：对行进行伸展，这样每行都可以有相等的长度。
（5）char：将内容对准指定字符。

valign属性控制表格单元格内容在垂直方向上的对齐方法，可以设置的值有以下几种。

（1）top：对内容进行上对齐。

（2）middle：默认对齐方式，对内容进行居中对齐。
（3）bottom：对内容进行下对齐。
（4）baseline：与基线对齐。

如果设置单元格的align属性为justify，单元格中的内容会以两端对齐的方式展现，但是一般情况下很少用到；而align属性值为char的效果，目前几乎没有浏览器支持；valign属性值为baseline时，只有单元格的内容为字母，且字号各不相同时效果才会明显，实际应用中也很少使用。

align和valign属性可以在<tr>、<th>或<td>标签中进行设置。当用于<tr>标签时，对齐属性应用于该行所有的单元格。以下代码列举了这两种属性在单元格中的详细使用情况，效果如图5.5所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>5.4</title>
</head>
<body>
<table border="1" cellspacing="0" width="300px" height="200px">
    <tr align="left" valign="top">
        <th>列头</th>
        <th>列头</th>
        <th>列头</th>
    </tr>
    <tr>
        <td align="left" valign="top">数据</td>
        <td align="center" valign="middle">数据</td>
        <td align="right" valign="bottom">数据</td>
    </tr>
    <tr>
        <td>数据</td>
        <td>数据</td>
        <td>数据</td>
    </tr>
</table>
</body>
</html>
```



图5.5

5.2.5 表格边框

表格的边框可以用border属性进行控制，border属性会为每个单元格应用边框，并用边框围绕表格。如果border属性的值发生改变，那么只有表格周围边框的尺寸会发生变化。在HTML页面中插入表格时，只有设置了表格的边框属性border，才能显示插入的表格，例如以下代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>5.5</title>
</head>
<body>
<table border="1px" width="300px" height="200px">
    <tr>
        <th>列头</th>
        <th>列头</th>
        <th>列头</th>
    </tr>
    <tr>
        <td>数据</td>
        <td>数据</td>
        <td>数据</td>
    </tr>
</table>
</body>
</html>
```

在这段代码中，我们设置了表格的border属性为1px，又设置了表格的宽度和高度，效果如图5.6所示。

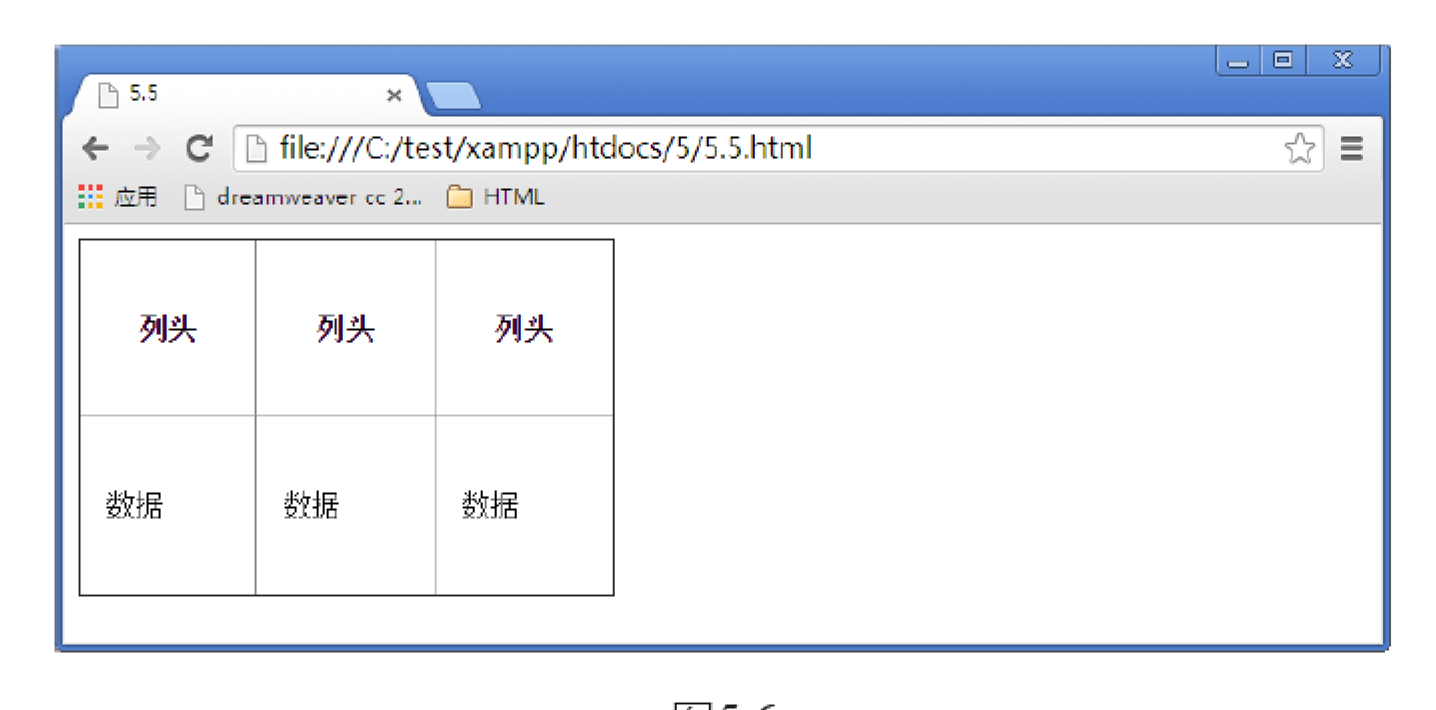


图5.6

5.2.6 单元格间距和单元格边距

仔细观察表格的边框，我们可以看到单元格之间、单元格与表格边框之间都有一个间隙，这个间隙被称为单元格间距，可以通过cellspacing来设置单元格的间距。例如设置单元格

的间距为0，可以让表格的边框看起来更窄，相关代码如下：

```
<table border="1px" width="300px" height="200px" cellspacing="0">
```

添加以上代码后，表格的效果如图5.7所示。



图5.7

目前看起来表格的边框有些粗，并不像我们设置的1个像素那么宽，这是因为虽然我们设置了表格的边框为1个像素，但是这里显示的却是两个<td>的边框。因为<td>之间并没有重合，要想看到1个像素边框的效果，需要设置表格的border-collapse属性为collapse，例如下面的代码：

```
<table border="1px" width="300px" height="200px" cellspacing="0"
style="border-collapse:collapse">
```

添加以上代码后，表格的边框就显示为真正的1个像素宽度，效果如图5.8所示。

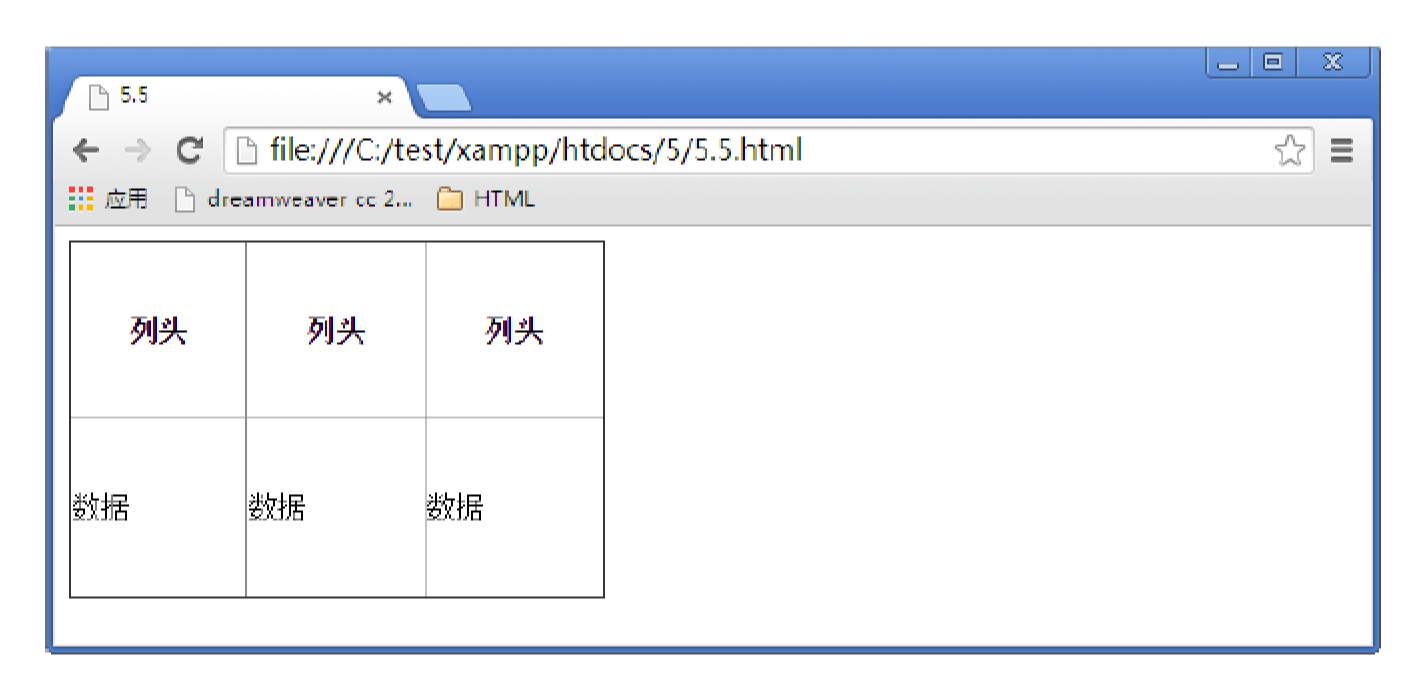


图5.8

单元格的边距表示单元格的内容与单元格边的距离，可以使用cellpadding属性进行设置。例如添加以下代码，设置数据单元格边距为15px，单元格中的文本将与边框保持15个像素的距离，效果如图5.9所示。

```
<table border="1px" width="300px" height="200px" cellpadding="15px"
cellspacing="0" style="border-collapse:collapse">
```

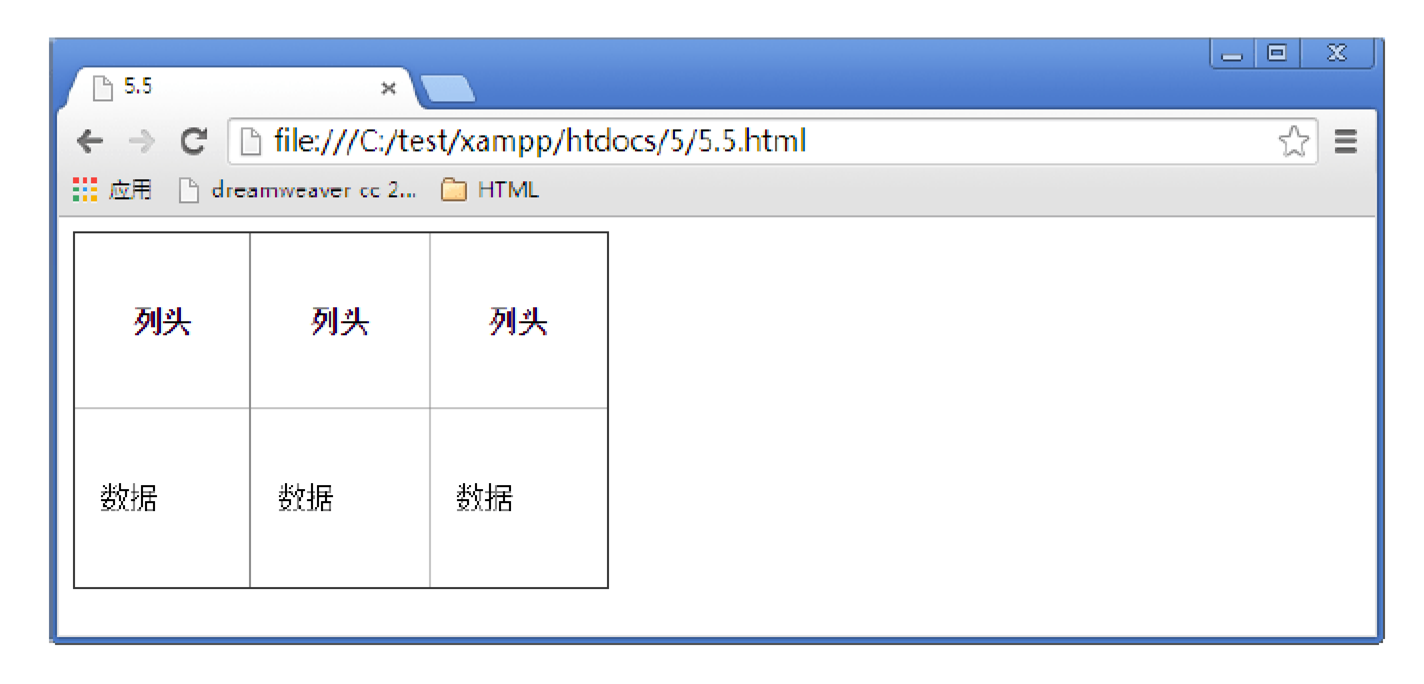


图5.9

5.2.7 表头

表格中的表头用<th>标签来定义，多数浏览器会将表头显示为粗体居中的文本，这些效果我们在前面的案例中已经看到了。通常情况下，表头的排列方式分两种，一种是水平显示在表格的第一行，相关代码如下所示，效果如图5.10所示。

```
    <table border="1px" width="300px" height="200px" cellspacing="0"
style="border-collapse:collapse">
        <tr>
            <th>表头</th>
            <th>表头</th>
            <th>表头</th>
        </tr>
        <tr>
            <td>数据</td>
            <td>数据</td>
            <td>数据</td>
        </tr>
    </table>
```

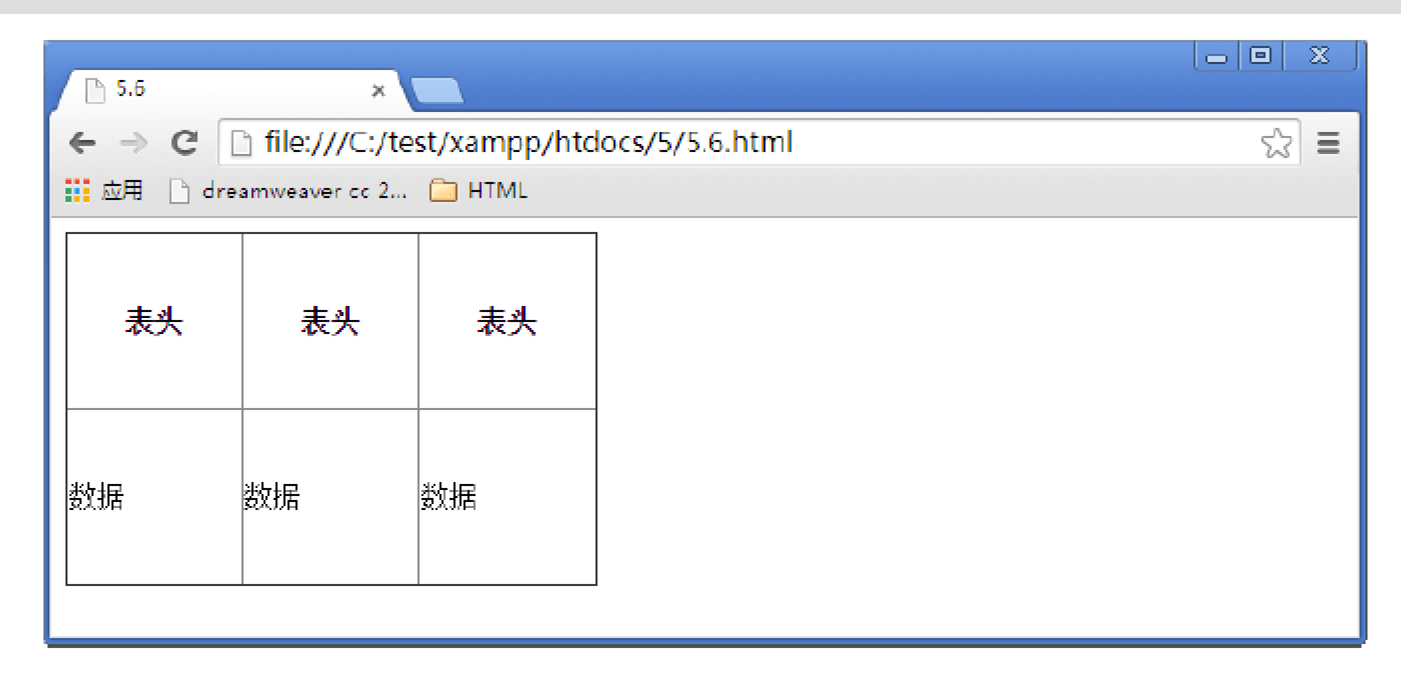


图5.10

另一种是垂直显示在表格的第一列，相关代码如下所示，效果如图5.11所示。

```
    <table border="1px" width="300px" height="200px" cellspacing="0"
style="border-collapse:collapse">
        <tr>
```

```
            <th>表头</th>
            <td>数据</td>
            <td>数据</td>
        </tr>
        <tr>
            <th>表头</th>
            <td>数据</td>
            <td>数据</td>
        </tr>
    </table>
```

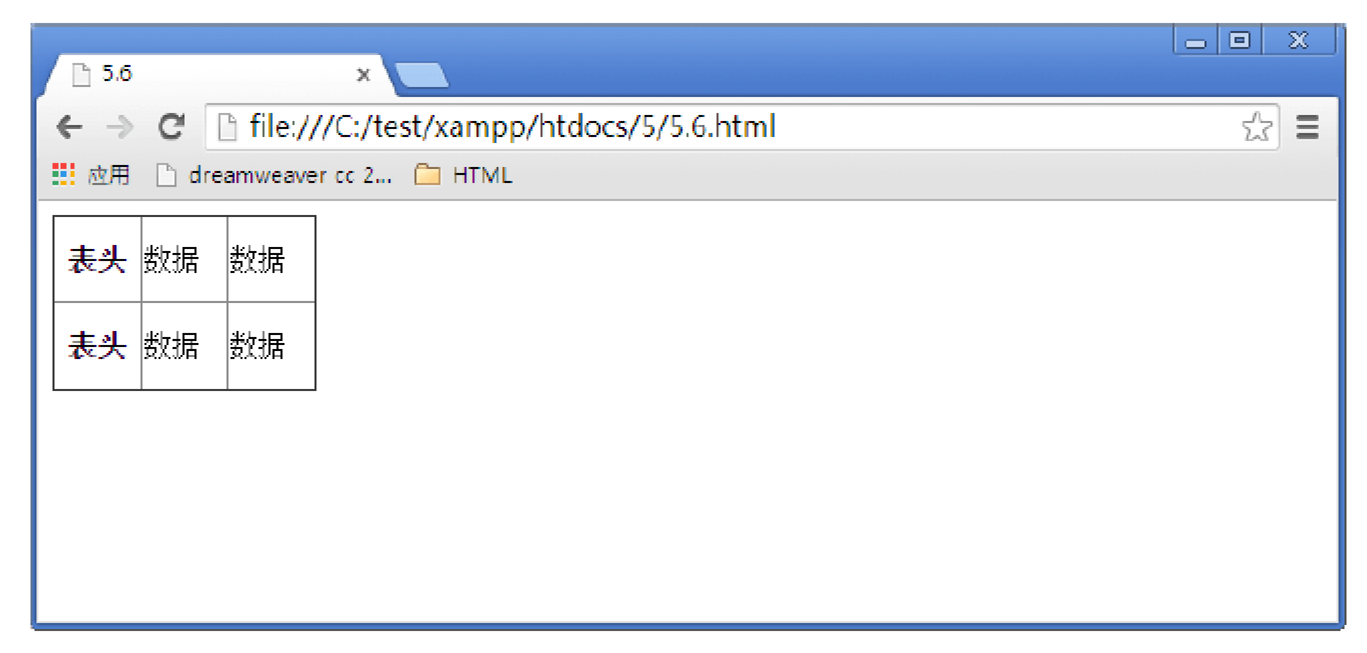


图5.11

更加复杂的表头需要用到表格的嵌套，以及colspan和rowspan属性，我们将在5.2.9节进行详细介绍。

5.2.8 nowrap属性

nowrap属性用于控制单元格中的内容是否自动换行，当nowrap属性值为true时表示允许换行，当nowrap属性值为false时表示禁止换行。需要注意的是，nowrap属性仅在没有设置width属性时起作用，如果设置了单元格的width属性，nowrap属性就没有效果了。

5.2.9 colspan和rowspan属性

colspan和rowspan属性能够改变表格的外观，使某列或者行合并起来。这两个属性作用于<td>标签上，colspan属性可以对任意行进行设置，使其跨越多列，而rowspan可以对第一列进行设置，使其跨越多行。

例如，以下代码是colspan跨越多列的效果，如图5.12所示。

```
    <!doctype html>
    <html>
    <head>
    <meta charset="utf-8">
    <title>5.7</title>
    </head>
    <body>
    <table border="1px" width="300px" height="200px" cellspacing="0"
style="border-collapse:collapse">
```

```
        <tr>
            <th colspan="2">表头</th>
            <th>表头</th>
        </tr>
        <tr>
            <td>数据</td>
            <td>数据</td>
            <td>数据</td>
        </tr>
    </table>
    </body>
    </html>
```



图5.12

以下代码是rowspan跨越多行的效果，如图5.13所示。

```
    <!doctype html>
    <html>
    <head>
    <meta charset="utf-8">
    <title>5.7</title>
    </head>
    <body>
    <table border="1px" width="300px" height="200px" cellspacing="0"
style="border-collapse:collapse">
        <tr>
            <th rowspan="2">表头</th>
            <td>数据</td>
            <td>数据</td>
        </tr>
        <tr>
            <td>数据</td>
            <td>数据</td>
        </tr>
    </table>
    </body>
    </html>
```



图5.13

5.2.10 背景与边框颜色

在使用表格展示数据的时候，我们经常会为表格的背景和边框设置颜色，以便用户能更清晰的观察数据。在HTML页面中，可以使用bgcolor属性设置颜色。如果要为整个表格设置同一个背景色，可以为<table>标签设置颜色属性；如果要为某一行设置背景色，可以为<tr>标签设置颜色属性；如果要为某个单元格设置颜色属性，可以直接为<th>或者<td>标签设置颜色属性。表格边框的颜色可以通过bordercolor属性进行设置。例如下面的代码：

```
    <!doctype html>
    <html>
    <head>
    <meta charset="utf-8">
    <title>5.8</title>
    </head>
    <body>
    <table border="1px" bordercolor="#F30F13" width="300px" height="200px"
cellspacing="0" style="border-collapse:collapse;color:#FDFDFD" >
        <tr bgcolor="#006699">
            <th>表头</th>
            <th>表头</th>
            <th>表头</th>
        </tr>
        <tr bgcolor="#B6B6B6">
            <td>数据</td>
            <td>数据</td>
            <td>数据</td>
        </tr>
        <tr bgcolor="#D89898">
            <td>数据</td>
            <td>数据</td>
            <td>数据</td>
        </tr>
    </table>
    </body>
    </html>
```

在这段代码中，使用bordercolor属性为表格设置了边框颜色，使用bgcolor属性为每行数据设置了背景颜色，效果如图5.14所示。

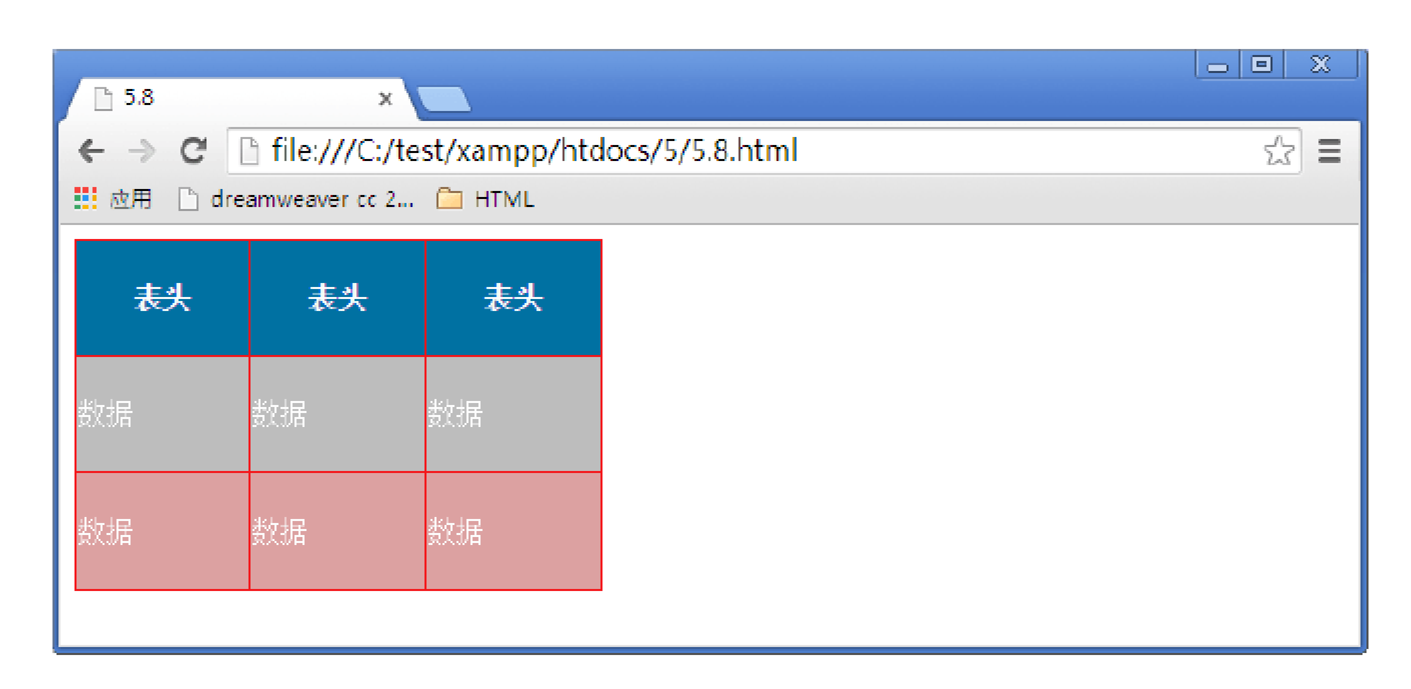


图5.14

5.2.11 背景图像

在HTML页面中可以使用background属性为表格添加背景图像。可以为整个表格添加统一的背景图像，也可以为单独某一行数据添加统一的背景图像，甚至可以为某个指定的单元格添加单独的背景图像。例如下面这段代码，分别为整个表格、第1行以及第2行第1列单元格添加了背景图像，效果如图5.15所示。

```
    <!doctype html>
    <html>
    <head>
    <meta charset="utf-8">
    <title>5.9</title>
    </head>
    <body>
    <table border="1px" width="300px" height="200px" cellspacing="0"
style="border-collapse:collapse; background:url(img/img04.png)" >
        <tr style="background:url(img/img02.png)">
            <th>表头</th>
            <th>表头</th>
            <th>表头</th>
        </tr>
        <tr>
            <td background="img/img03.png">数据</td>
            <td>数据</td>
            <td>数据</td>
        </tr>
    </table>
    </body>
    </html>
```



图5.15



5.3 制作表格

在HTML页面设计中，表格经常被用于展示数据和页面布局。随着Web技术的不断发展，表格的应用已经越来越少，但是在某些情况下，我们依然需要使用表格来满足设计的要求。本节将着重介绍如何制作如图5.16所示的表格。

5.9-1
file:///C:/xampp/htdocs/5/5.9-1.html

2015年部门评分表

部门名称	2015年											
	1月	2月	3月	4月	5月	6月	7月	8月	9月	10月	11月	12月
生产部	78	85	88	86	83	79	89	90	92	91	92	93
品管部	80	85	88	87	89	90	92	91	89	95	96	94
销售部	70	75	65	68	90	95	98	69	86	88	89	92
市场部	60	65	88	85	95	98	97	88	89	90	91	95
统计	78	88	87	85	90	82	78	88	90	85	90	92

图5.16

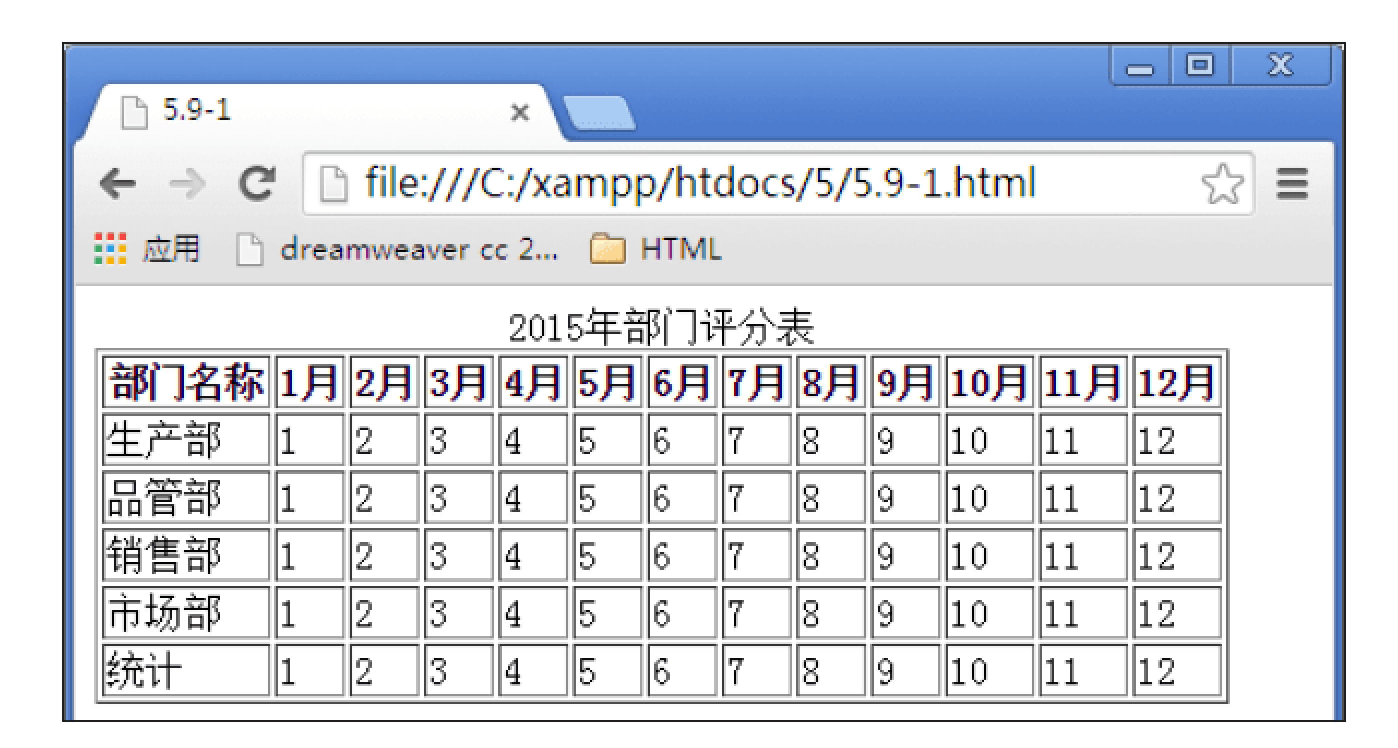
01 新建一个HTML页面，在body元素中嵌入一个table元素，实现基本的行列功能。设置表格的caption为“2015年部门评分表”，thead标签表示表头为“部门名称”和12个月份，tfoot标签表示部门统计结果，相关代码如下：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>5.9-1</title>
</head>
<body>
<table border="1">
```

```
    <caption>2015年部门评分表</caption>
    <thead>
     <tr>
      <th>部门名称</th>
      <th>1月</th>
      <th>2月</th>
      <th>3月</th>
      <th>4月</th>
      <th>5月</th>
      <th>6月</th>
      <th>7月</th>
      <th>8月</th>
      <th>9月</th>
      <th>10月</th>
      <th>11月</th>
      <th>12月</th>
     </tr>
    </thead>
    <tfoot>
     <tr>
      <td>统计</td>
      <td>1</td>
      <td>2</td>
      <td>3</td>
      <td>4</td>
      <td>5</td>
      <td>6</td>
      <td>7</td>
      <td>8</td>
      <td>9</td>
      <td>10</td>
      <td>11</td>
      <td>12</td>
     </tr>
    </tfoot>
    <tbody>
     <tr>
      <td>生产部</td>
      <td>1</td>
      <td>2</td>
      <td>3</td>
      <td>4</td>
      <td>5</td>
      <td>6</td>
      <td>7</td>
      <td>8</td>
      <td>9</td>
      <td>10</td>
      <td>11</td>
      <td>12</td>
     </tr>
```

```
    <tr>
     <td>品管部</td>
     <td>1</td>
     <td>2</td>
     <td>3</td>
     <td>4</td>
     <td>5</td>
     <td>6</td>
     <td>7</td>
     <td>8</td>
     <td>9</td>
     <td>10</td>
     <td>11</td>
     <td>12</td>
    </tr>
    <tr>
     <td>销售部</td>
     <td>1</td>
     <td>2</td>
     <td>3</td>
     <td>4</td>
     <td>5</td>
     <td>6</td>
     <td>7</td>
     <td>8</td>
     <td>9</td>
     <td>10</td>
     <td>11</td>
     <td>12</td>
    </tr>
    <tr>
     <td>市场部</td>
     <td>1</td>
     <td>2</td>
     <td>3</td>
     <td>4</td>
     <td>5</td>
     <td>6</td>
     <td>7</td>
     <td>8</td>
     <td>9</td>
     <td>10</td>
     <td>11</td>
     <td>12</td>
    </tr>
   </tbody>
   </table>
   </body>
   </html>
```

在浏览器中浏览这段代码的效果，如图5.17所示。



2015年部门评分表

部门名称	1月	2月	3月	4月	5月	6月	7月	8月	9月	10月	11月	12月
生产部	1	2	3	4	5	6	7	8	9	10	11	12
品管部	1	2	3	4	5	6	7	8	9	10	11	12
销售部	1	2	3	4	5	6	7	8	9	10	11	12
市场部	1	2	3	4	5	6	7	8	9	10	11	12
统计	1	2	3	4	5	6	7	8	9	10	11	12

图5.17

02 下面我们对表头进行修改，除了第1行第1列外，将其他表头拆分成两行，并将拆分后的第1行合并，相关代码如下所示：

```
<thead>
 <tr>
    <th rowspan="2">部门名称</th>
    <th colspan="12">2015年</th>
 </tr>
 <tr>
    <th>1月</th>
    <th>2月</th>
    <th>3月</th>
    <th>4月</th>
    <th>5月</th>
    <th>6月</th>
    <th>7月</th>
    <th>8月</th>
    <th>9月</th>
    <th>10月</th>
    <th>11月</th>
    <th>12月</th>
 </tr>
</thead>
```

刷新浏览器，效果如图5.18所示。

5.9-1
file:///C:/xampp/htdocs/5/5.9-1.html
应用 dreamweaver cc 2... HTML

2015年部门评分表

部门名称	2015年											
	1月	2月	3月	4月	5月	6月	7月	8月	9月	10月	11月	12月
生产部	1	2	3	4	5	6	7	8	9	10	11	12
品管部	1	2	3	4	5	6	7	8	9	10	11	12
销售部	1	2	3	4	5	6	7	8	9	10	11	12
市场部	1	2	3	4	5	6	7	8	9	10	11	12
统计	1	2	3	4	5	6	7	8	9	10	11	12

图5.18

03 表格的基本结构已经形成，但是这样的表格看起来很丑，我们通过设置表格的宽度、高度、边框、对齐方式、背景色等属性，对表格进行美化，相关代码如下：

```
<style>
*{margin:0 auto}
table{
font-size:12px;
width:800px;
}
table caption{font-size:24px;}
table tr td{
padding:5px;
background-color:#D6E9F0;
text-align:center;
}
table tr th{
padding:5px;
background-color:#D6E9F0;
}
.thmain{background-color:#66A9BC;}
.tdmain{background-color:#91C5D3; text-align:left;}
.tdcount{background-color:#B0CC7F;}
.tdresult{background-color:#D7E1C8; text-align:center;}
</style>
…
<th class="thmain" rowspan="2">部门名称</th>
<th class="thmain" colspan="12">2015年</th>
…
<tfoot>
 <tr>
  <td class="tdcount">统计</td>
  <td class="tdresult">1</td>
  <td class="tdresult">2</td>
  <td class="tdresult">3</td>
  <td class="tdresult">4</td>
  <td class="tdresult">5</td>
  <td class="tdresult">6</td>
  <td class="tdresult">7</td>
  <td class="tdresult">8</td>
  <td class="tdresult">9</td>
  <td class="tdresult">10</td>
  <td class="tdresult">11</td>
  <td class="tdresult">12</td>
 </tr>
</tfoot>
…
<td class="tdmain">生产部</td>
…
<td class="tdmain">品管部</td>
…
```

```
    <td class="tdmain">销售部</td>
    …
    <td class="tdmain">市场部</td>
```

刷新页面后的效果如图5.19所示。

2015年部门评分表

部门名称	2015年											
	1月	2月	3月	4月	5月	6月	7月	8月	9月	10月	11月	12月
生产部	1	2	3	4	5	6	7	8	9	10	11	12
品管部	1	2	3	4	5	6	7	8	9	10	11	12
销售部	1	2	3	4	5	6	7	8	9	10	11	12
市场部	1	2	3	4	5	6	7	8	9	10	11	12
统计	1	2	3	4	5	6	7	8	9	10	11	12

图5.19

04 修改页面中的数据，最终结果如图5.16所示。



5.4 插入列表

列表是HTML中另一个用于组织内容的工具，结合CSS的使用，它不仅可以制作各种功能菜单，还可以用于布局页面内容。列表的种类有3种，分别为有序列表、无序列表和定义列表，下面我们就详细介绍这3种列表的使用方法。

5.4.1 有序列表

有序列表用<ol>标签表示，每个列表项用<li>标签表示，这样组织的内容在每一个列表项前面都有一个序号，例如下面的代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>5.10</title>
</head>
<body>
<ol>
    <li>项目</li>
    <li>项目</li>
    <li>项目</li>
</ol>
```

```
</body>
</html>
```

在这段代码中，有序列表中有3个<li>元素，每个元素的内容都一样，且标签前面没有序号，但是在页面浏览的时候，这些标签前面就会出现序号，如图5.20所示，这就是有序列表的一个明显特征。

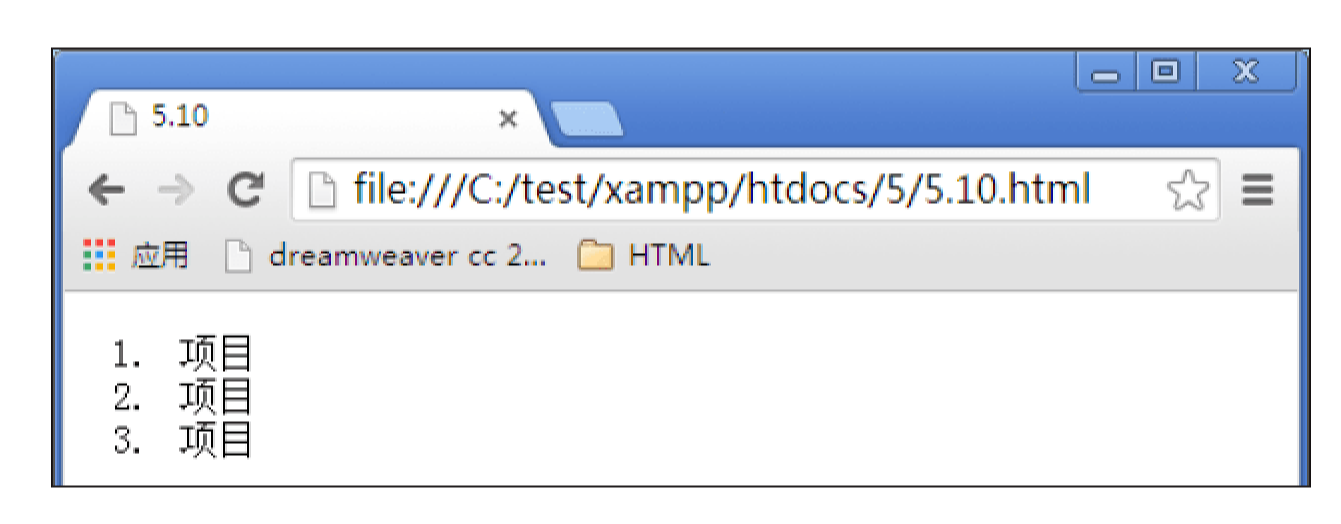


图5.20

默认情况下，有序列表以数字为序号，我们还可以通过设置它的type属性修改序号类型.例如使用<ol type="a">修改序号为小写字母，使用<ol type="A">修改序号为大写字母，使用<ol type="i">修改序号为小写罗马数字，使用<ol type="I">修改序号为大写罗马数字。另外通过设置<ol>标签的start属性，还可以设置有序列表的起始序号。例如，<ol start="5">表示有序列表的序号从5开始。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>5.10</title>
</head>
<body>
<h2>数字序号列表</h2>
<ol type="1">
    <li>项目</li>
    <li>项目</li>
    <li>项目</li>
</ol>
<h2>小写字母序号列表</h2>
<ol type="a">
    <li>项目</li>
    <li>项目</li>
    <li>项目</li>
</ol>
<h2>大写字母序号列表</h2>
<ol type="A">
    <li>项目</li>
    <li>项目</li>
    <li>项目</li>
</ol>
<h2>小写罗马数字序号列表</h2>
<ol type="i">
```

```
        <li>项目</li>
        <li>项目</li>
        <li>项目</li>
    </ol>
    <h2>大写罗马数字序号列表</h2>
    <ol type="I">
        <li>项目</li>
        <li>项目</li>
        <li>项目</li>
    </ol>
    <h2>从5开始的数字序号列表</h2>
    <ol start="5">
        <li>项目</li>
        <li>项目</li>
        <li>项目</li>
    </ol>
    </body>
    </html>
```

运行这段代码后的效果如图5.21所示。



图5.21

5.4.2 无序列表

有序列表相对应的是无序列表，无序列表没有具体的序列号，是用一些符号进行标记的。无序列表用<ul>标签表示，每个列表项目仍然用<li>标签表示。无序列表可用3种符号进行标记，默认情况下以实心圆进行标记，我们还可以通过设置<ul>标签的type属性指定其他的符号。例如，<ul type="circle">表示空心圆列表，<ul type="square">表示实心方块列表，相关代码如下所示：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>5.11</title>
</head>
<body>
<h2>实心圆列表</h2>
<ul type="disc">
<li>项目</li>
    <li>项目</li>
    <li>项目</li>
</ul>
<h2>空心圆列表</h2>
<ul type="circle">
<li>项目</li>
    <li>项目</li>
    <li>项目</li>
</ul>
<h2>实心方块列表</h2>
<ul type="square">
<li>项目</li>
    <li>项目</li>
    <li>项目</li>
</ul>
</body>
</html>
```

运行这段代码后的效果如图5.22所示。

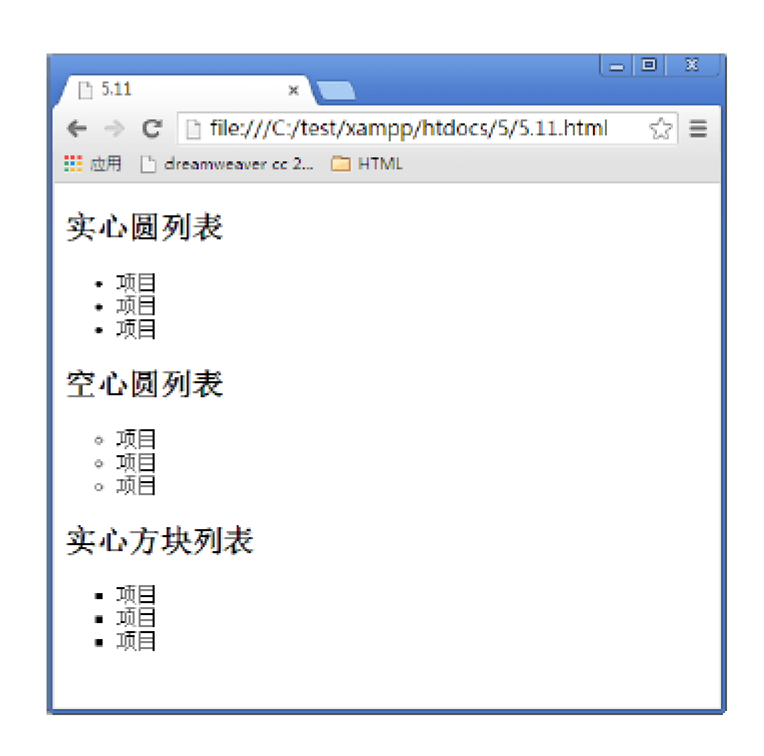


图5.22

5.4.3 定义列表

在HTML页面中，还有一种列表称之为定义列表，它的列表项是由项目及其注释组成的。定义列表用<dl>标签表示，其中又包含<dt>标签和<dd>标签，<dt>标签用于定义列表项目，而<dd>标签用于对项目进行注释。一个<dl>标签中可以包含多组项目标签和注释。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>5.12</title>
</head>
<body>
<dl>
    <dt>标题</dt>
        <dd>内容</dd>
        <dd>内容</dd>
    <dt>智能手机</dt>
    <dd>苹果手机</dd>
        <dd>三星手机</dd>
        <dd>华为手机</dd>
</dl>
</body>
</html>
```

运行这段代码后的效果如图5.23所示。

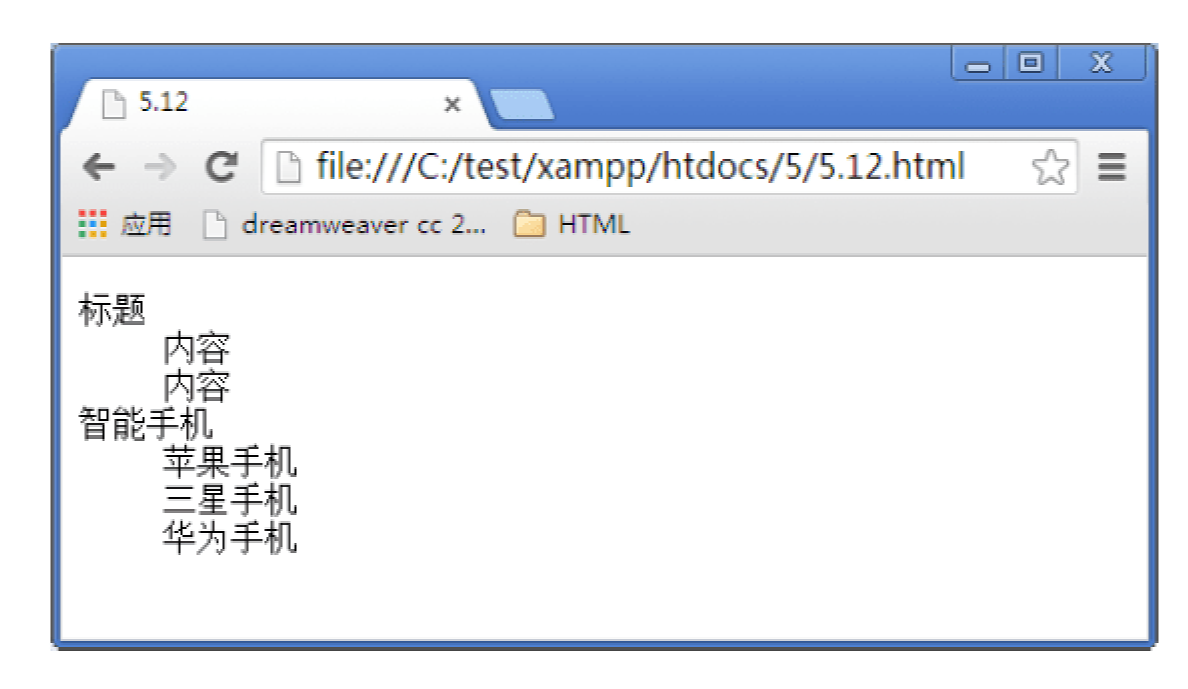


图5.23



5.5 制作横向导航

列表除了用于组织HTML内容以外，还经常与CSS一起用于制作各种导航效果。本节将介绍如何使用无序列表和CSS制作横向导航条。

01 新建一个HTML页面，在代码视图中编写如下代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
```

```
<title>5.13</title>
</head>
<body>
<div id="nav">
    <ul>
    <li><a href="#">首页 | </a></li>
        <li><a href="#">集团概况 | </a></li>
        <li><a href="#">品牌中心 | </a></li>
        <li><a href="#">产品中心 | </a></li>
        <li><a href="#">项目优势 | </a></li>
        <li><a href="#">加盟我们 | </a></li>
        <li><a href="#">新闻中心 | </a></li>
        <li><a href="#">联系我们</a></li>
    </ul>
</div></body>
</html>
```

这段代码在浏览器中的显示效果如图5.24所示。

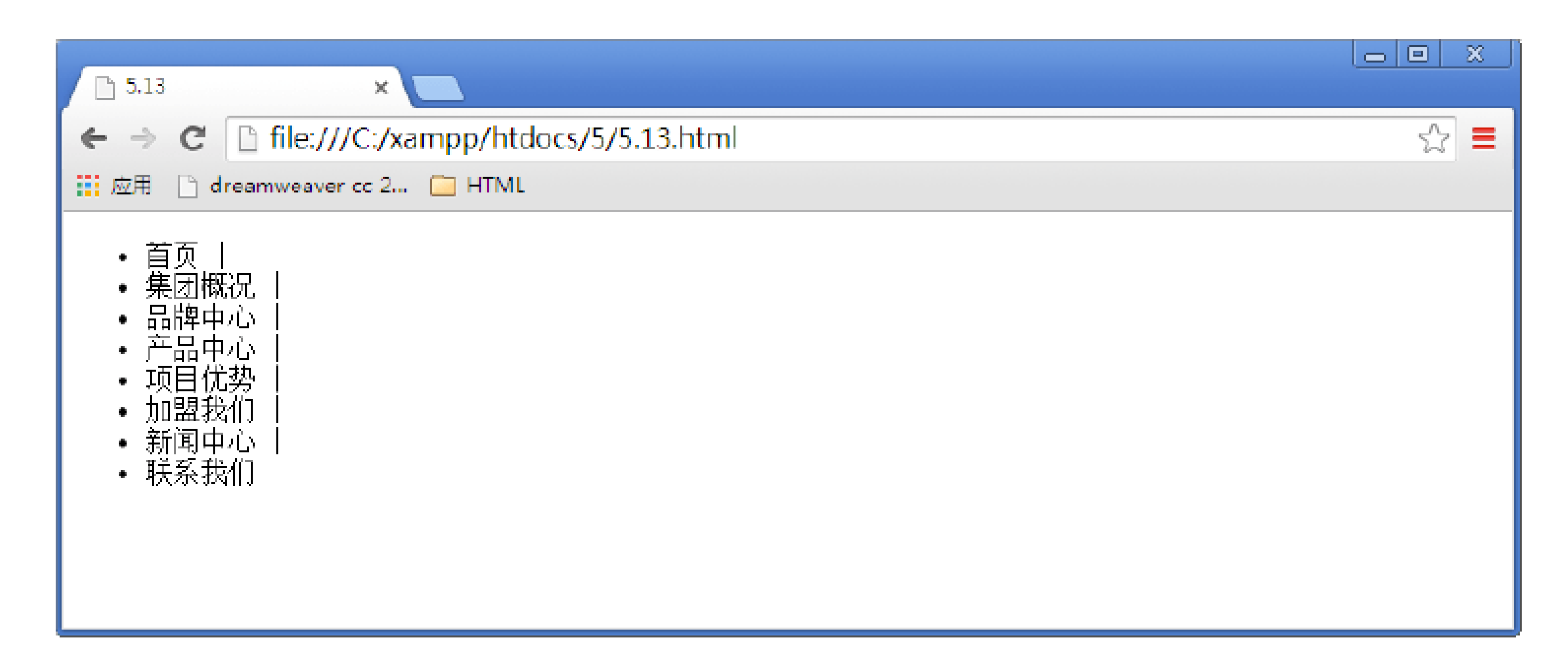


图5.24

02 设置无序列表的CSS样式，去掉列表前的小圆点，相关代码如下所示：

```
<style>
#nav ul{
list-style:none;
}
</style>
```

刷新浏览器后的效果如图5.25所示。

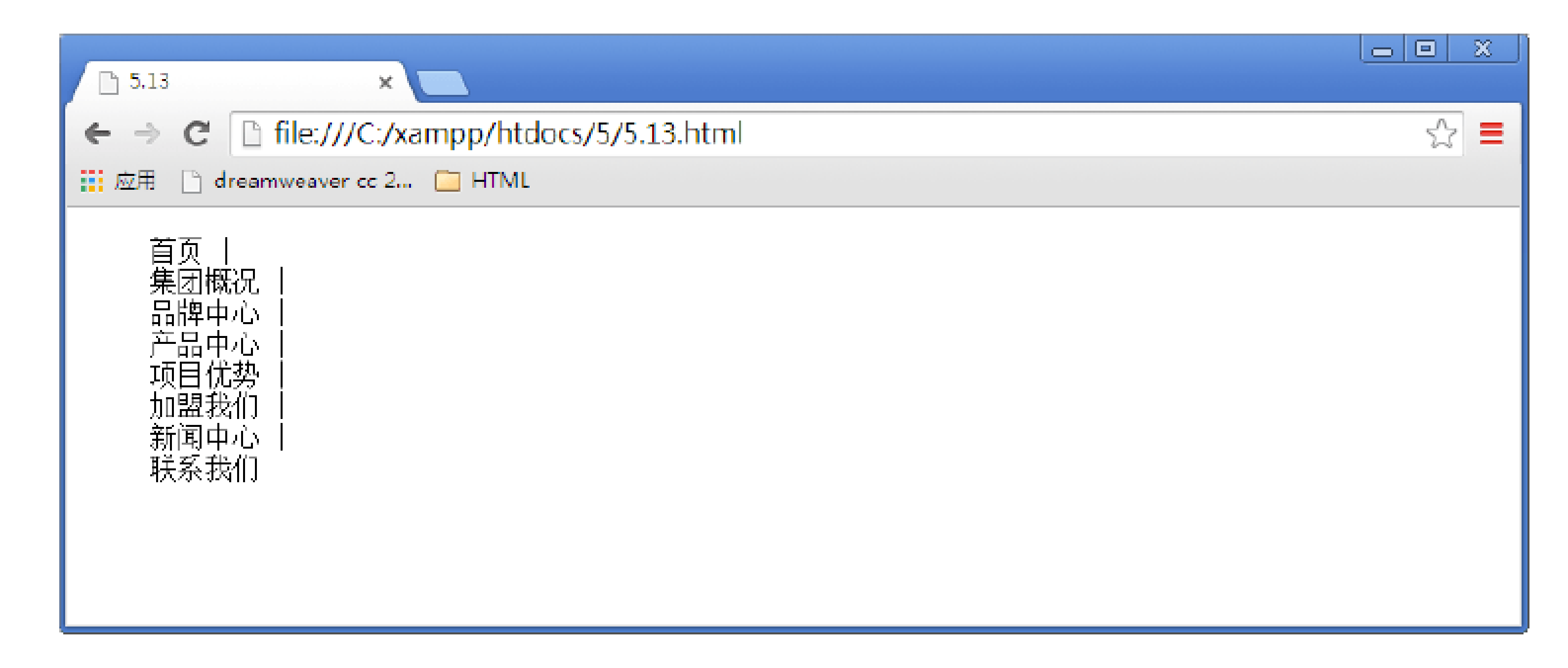


图5.25

03 设置列表项横向浮动，并设置字体颜色，相关代码如下所示：

```
<style>
a{
text-decoration:none;
font-size:20px;
color:white;
padding:10px;
}
#nav ul{
list-style:none;
}
#nav li{
float:left;
line-height:50px;
background-color:#1E1E1E;
font-family:'微软雅黑';
}
</style>
```

刷新浏览器后的效果如图5.26所示。



图5.26

第6章 CSS基础

HTML标签和CSS样式是组成网页的两个最重要的元素，如果说HTML标签是组成网页的框架，那么CSS样式就是网页框架外华丽的外衣。本章我们就来详细介绍CSS的基础知识，掌握如何让网页以最漂亮的形式展现给大家。



6.1 认识CSS样式表

早期的HTML页面为了做出漂亮的界面，就为HTML增加了很多属性，这样做的结果就是页面代码变得十分臃肿，而且不利于维护。CSS在解读这一问题中承担了重要的角色。

6.1.1 CSS是什么

CSS是Cascading Style Sheets的简称，也叫做层叠样式表，是一种用来表现HTML、XML等文件样式的计算机语言。

6.1.2 CSS能做什么

我们形象地将CSS比喻为HTML页面华丽的外衣，如果一个HTML页面使用两套不同的CSS样式，就能呈现出两种截然不同的风格。例如下面这段HTML代码，页面的主要内容是3个div元素，没有使用任何CSS样式的时候，运行这段代码后，我们可以看到如图6.1所示页面效果。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.1</title>
</head>
<body>
<div id="box1">内容1</div>
<div id="box2">内容2</div>
<div id="box3">内容3</div>
</body>
</html>
```

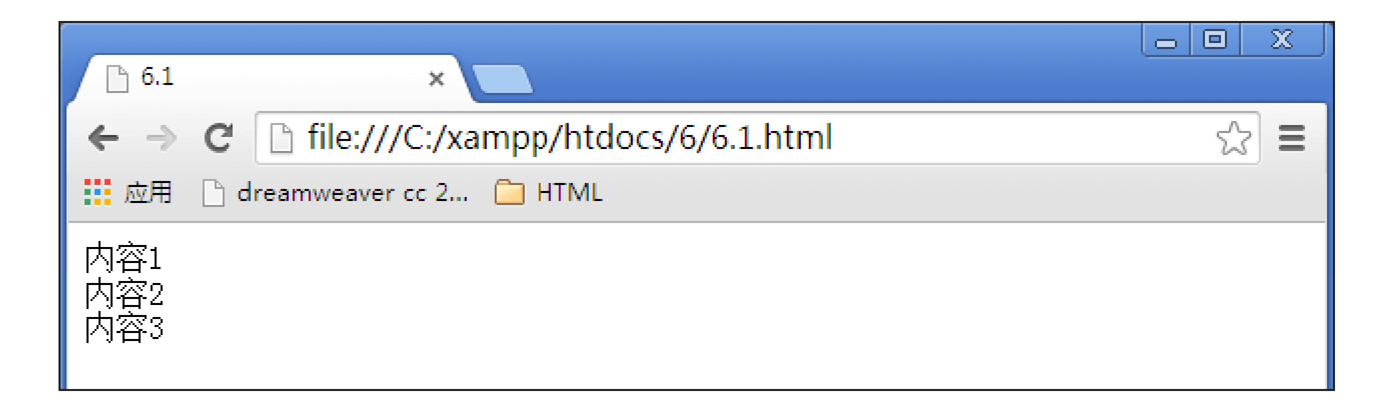


图6.1

如果使用了以下CSS样式，运行后的效果如图6.2所示。

```
<style>
div{
width:80px;
height:60px;
font-size:20px;
font-weight:bold;
margin-top:5px;
border:solid 1px red;
}
#box1{
background-color:#DFCDA2;
color:#C85E72;
}
#box2{
background-color:#A1E7CE;
color:#3D6212;
}
#box3{
background-color:#F0ABEB;
color:#06606F;
}
</style>
```

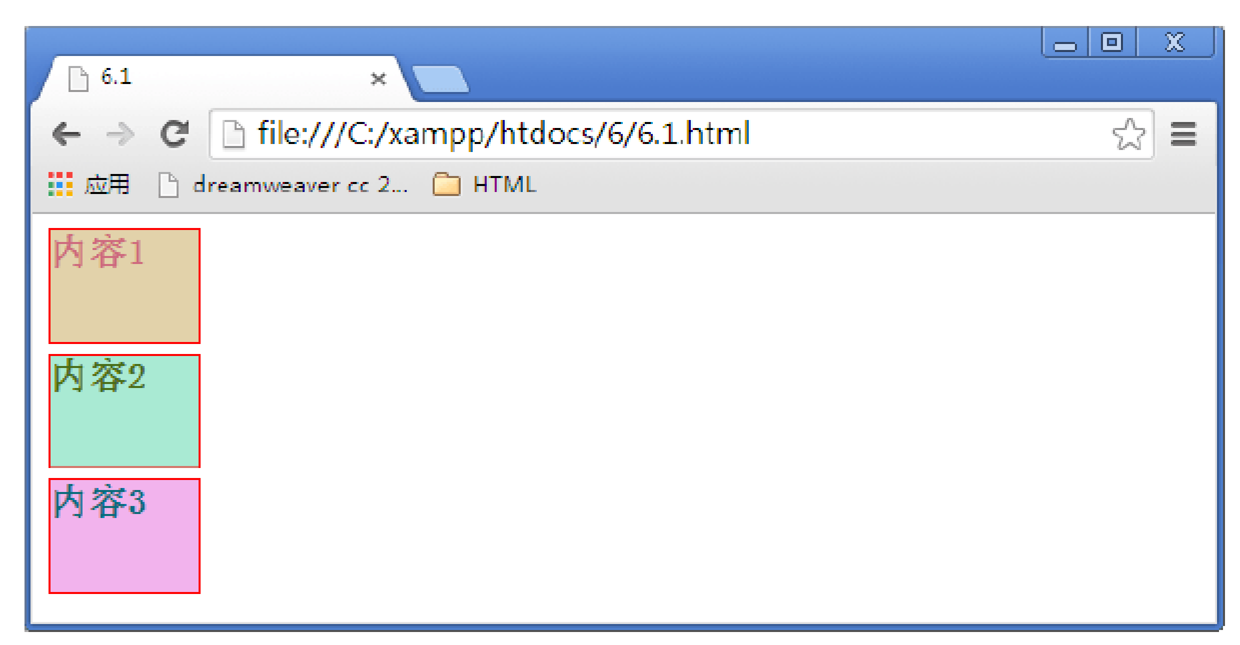


图6.2

如果使用了以下CSS样式，运行后的效果如图6.3所示。

```
<style>
div{
width:80px;
height:60px;
```

```
    font-size:20px;
    font-weight:bold;
    text-align:center;
    border:solid 1px red;
    float:left;
    margin-left:5px;
    line-height:60px;
    }
    #box1{
    background-color:#84E5C8;
    color:#E80F0F;
    }
    #box2{
    background-color:#D49293;
    color:#009E12;
    }
    #box3{
    background-color:#E8D37B;
    color:#1729EC;
    }
    </style>
```

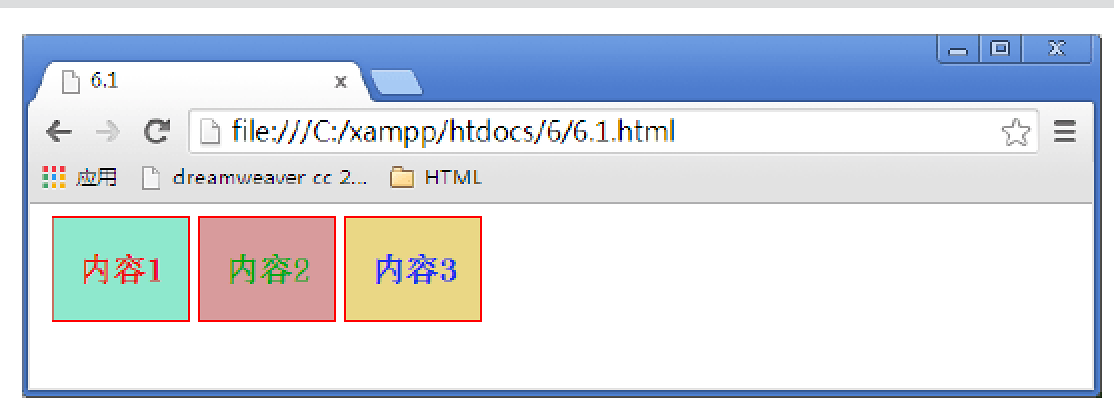


图6.3

从以上案例中我们可以看出，CSS可以控制网页中元素的布局、字体的大小和粗细、页面的宽度、各个元素的排列方式、背景等各种样式，CSS主要的作用就是美化网页。

6.1.3 CSS与HTML的区别

CSS与HTML都是组成网页的重要元素，它们各有各的功能，谁也不能替代对方。如果仅仅使用HTML标签堆叠出一个网页，那么这个网页将会极其丑陋，毫无美感可言，就好比建设中的摩天大楼，徒有耸入云霄的框架，却处处堆满建筑材料，狼藉不堪；相反，如果有多套设计精美的CSS样式，却没有一个合适的HTML页面来承载它们，那么这些CSS样式也不可能以最美的方式展示出来。

作为超文本标记语言的HTML用于组织网页的内容，而作为层叠样式表的CSS则用于控制HTML如何显示。HTML由众多的HTML标签组成，而CSS则通过层叠样式表控制这些HTML标签以什么样的样式进行展示。

6.1.4 CSS有哪些优势

CSS是网页风格的决定性因素，它的主要作用就是美化网页。相比只使用HTML自带的

样式，CSS可以实现更多的功能和样式，它的优势主要体现在以下几个方面：

（1）文档的结构和表现相分离，便于设计人员独立编写CSS样式。也便于后期维护。
（2）样式定义精确到像素的级别。
（3）浏览器支持多重样式，使网站同时支持多种表现形式。
（4）使用CSS的网页占用宽度比较少，浏览器支持CSS样式缓存，从而增强了网站的性能。
（5）使用样式表可以针对不同的设备类型对网站的内容进行优化。



6.2 CSS的工作原理

CSS通过设置属性来控制HTML的样式，许多CSS属性都与HTML属性相似，下面我们就来详细介绍CSS的基本语法以及CSS的类型。

6.2.1 CSS基本语法

CSS语法规则主要由两部分构成，一个是选择器，另一个是一条或多条声明。选择器通常是需要改变样式的HTML元素。每条声明则是由一个属性和一个值组成，属性与值之间用冒号分开，多个声明之间用分号隔开，声明必须用花括号围起来。例如下面的代码：

```
h1 {color:red;font-size:14px;}
```

其中选择器为“h1”，“color”和“font-size”都是属性，“red”和“14px”都是值。在设置颜色的值时，可以使用CSS允许的引文单词，如red、green、blue等，还可以使用十六进制颜色值，如下所示：

```
h1 {color:#ff0000}
```

或者使用CSS的缩写形式，如下所示：

```
h1 {color:#f00}
```

或者使用RGB颜色值，如下所示：

```
h1 {color:rgb(255,0,0)}
h1 {color:rgb(100%,0%,0%)}
```

另外，如果值为若干单词时，则需要给值加引号，如下所示：

```
p {font-family: "sans serif";}
```

6.2.2 CSS类型

根据CSS使用位置的不同，可以将CSS的类型分为3类，分别是外部样式表、内部样式表

和行间样式表。外部样式表是指将CSS样式单独作为一个样式文件，在HTML文档中引入；内部样式表是指将CSS样式作为HTML页面中的一部分，写在<style>标签中；而行间样式表是指将CSS样式与HTML属性写在一起的一种方式，这种方式可在测试时使用，通常情况下不建议使用。

6.3 CSS样式的引用方法

1. 外部样式表

要使用外部样式表，首先需要创建一个CSS样式表文件，将CSS样式都写在这个文件中，然后就可以在HTML页面中使用<link>标签将外部样式表引入到文件中。例如，我们在名为CSS的文件夹中新创建一个名为index.css的CSS文件，而index.html文件与CSS文件夹在同一个目录下，这样我们就可以在index.html文件中使用<link>标签引入外部样式表，详细代码如下：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.2</title>
<link type="text/css" rel="stylesheet" href="CSS/index.css">
</head>
<body>
<h2>外部样式表</h2>
<ul>
    <li>项目</li>
    <li>项目</li>
    <li>项目</li>
    <li>项目</li>
</ul>
</body>
</html>
```

外部样式表通常会被多个文件使用。例如，网站上所有网页都有一个相同的区域，现在需要修改网页中这个区域的背景颜色，如果逐个修改网页背景色的CSS代码，那将是一件非常痛苦的事情。使用外部样式表后，因为所有网页都引入了这个样式，所以只需要修改外部样式表中对应区域的背景色，就可以修改所有页面的颜色。这里需要注意的是，<link>标签必须作为<head>标签的内容出现。

2. 内部样式表

在使用内部样式表时，我们不需要为CSS样式单独创建一个样式文件，可以直接在HTML页面中的<head>标签中使用<style>标签设置样式。例如下面的代码：

```
<!doctype html>
<html>
```

```
<head>
<meta charset="utf-8">
<title>6.2</title>
<style type="text/css">
h2 {
color:#7E2729;
}
ul{
list-style:none;
width:60px;
text-align:center;
}
li{
border:solid 1px #C0D3D4;
margin-top:2px;
background-color:#1B5164;
color:white;
}
</style>
</head>
<body>
<h2>外部样式表</h2>
<ul>
    <li>项目</li>
    <li>项目</li>
    <li>项目</li>
    <li>项目</li>
</ul>
</body>
</html>
```

在这段代码中，我们为HTML页面中的元素设置了3个CSS样式，这3个CSS样式位于<style>标签中，与<style>标签一起作为HTML页面中的一部分出现，其中type属性是指style元素以CSS的语法定义。需要注意的是，内部样式表只作用于当前的HTML页面。

3. 行间样式表

可以直接在HTML标签内通过设置style属性的值，设置各个HTML元素的样式。通过这种方法设置的样式表，只能作用于当前的HTML元素，例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.3</title>
</head>
<body>
<h2 style="color:#7E2729">外部样式表</h2>
<ul style="list-style:none; width:60px; text-align:center;">
        <li style="border:solid 1px #C0D3D4; margin-top:2px; background-color:#1B5164; color:white;">项目</li>
        <li style="border:solid 1px #C0D3D4; margin-top:2px; background-color:#1B5164; color:white;">项目</li>
        <li style="border:solid 1px #C0D3D4; margin-top:2px; background-color:#1B5164; color:white;">项目</li>
```

```
</ul>
</body>
</html>
```

这段代码的效果与内部样式表中代码的效果完全相同，但是这段代码中有很多的CSS样式都是重复的，而且HTML代码与CSS代码混杂在一起，很大程度上降低了代码的可读性。如果现在要替换项目列表的背景色，我们需要逐个修改项目列表的样式，这将是一件很麻烦的事情，所以不建议在项目中使用这种样式。



6.4 CSS选择器

选择器是CSS的核心，从最初的标签选择器、类选择器、id选择器到CSS3中提供的更丰富的选择器，定位页面上的任意元素开始变得越发简单。

6.4.1 标签选择器

标签选择器是最基本的CSS选择器，HTML文档中的元素本身就是一个选择器，例如要设置h1元素的字体颜色为红色，可以通过以下代码声明标签选择器：

```
<style>
h1 {
color: red;
}
</style>
```

如果要为h1设置更多的样式，可以继续在花括号中添加其他属性和值，例如下面这段代码：

```
<style>
h1 {
color: red;
font-size:24px;
}
</style>
```

6.4.2 class选择器

可以在HTML标签中指定class属性，将class属性的值作为class（类）选择器。一个HTML页面中，不同的HTML标签可以有相同的class属性值，应用相同class选择器的HTML标签具有相同的样式。例如下面这段代码：

```
<!doctype html>
<html>
<head>
```

```
<meta charset="utf-8">
<title>6.4</title>
<style type="text/css">
.redText{
color:red;
}
</style>
</head>
<body>
<h2 class="redText">标题</h2>
<p class="redText">段落中的文字</p>
</body>
</html>
```

在这段代码中，HTML页面中主要有两个元素，一个是h2标签，一个是p标签。这两个元素有一个共同的class属性值redText。在内部样式表中，定义了一个名为redText的样式，该样式中设置字体为红色。因为这两个元素使用了相同的类选择器，所以它们的字体颜色都是红色，运行这段代码后的效果如图6.4所示。

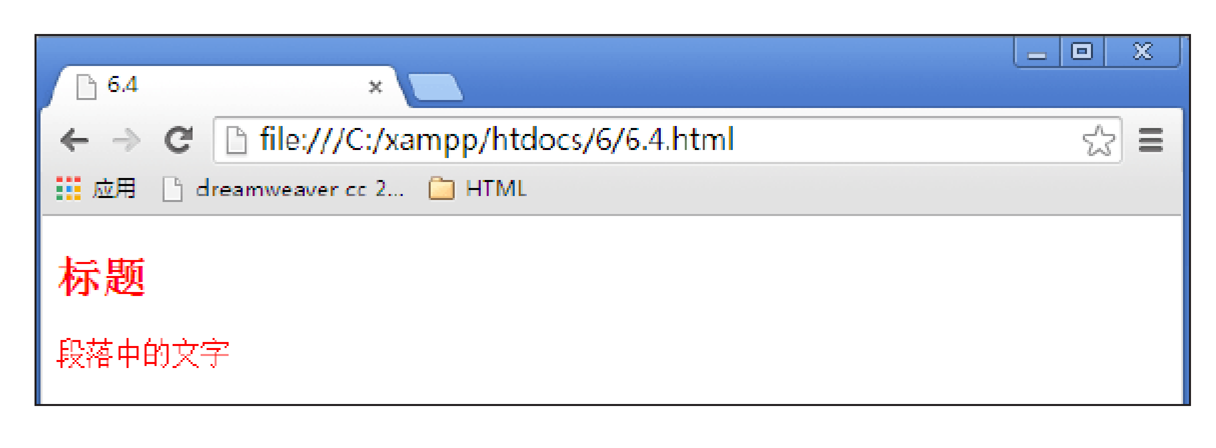


图6.4

6.4.3 id选择器

在HTML页面中，id属性确定了HTML元素的唯一性。通过id属性设置的样式称为id选择器，所以id选择器只能应用于页面中唯一确定的元素。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.5</title>
<style type="text/css">
#redText{
color:red;
}
#blueText{
color:blue;
font-weight:bold;
}
</style>
</head>
<body>
<h2 id="redText">标题</h2>
<p id="blueText">段落中的文字</p>
</body>
```

```
</html>
```

在这段代码中，h2元素的id属性值为“redText”，p元素的id属性值为“blueText”，分别为这两个元素设置id选择器，并设置它们的颜色为红色和蓝色，运行这段代码后的效果如图6.5所示。

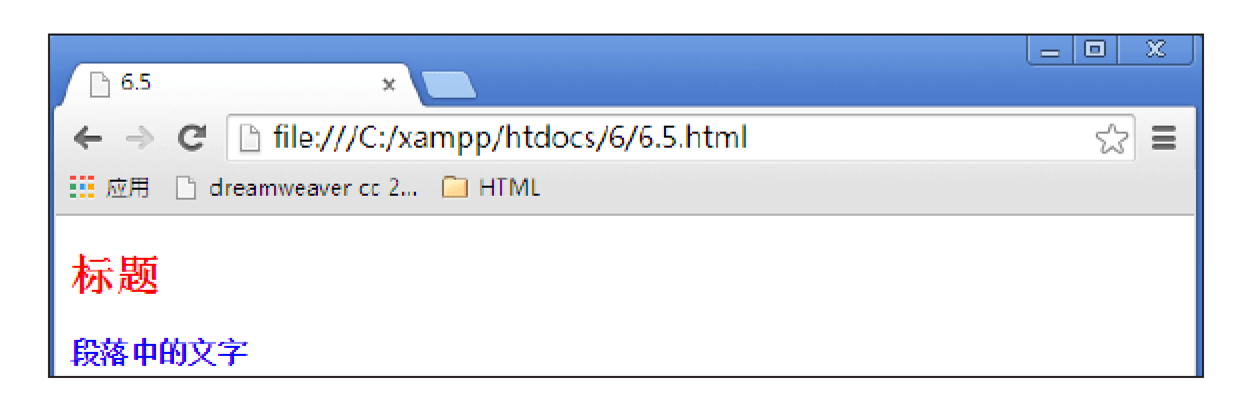


图6.5

6.4.4 通配符选择器

通配符用星号（*）表示，通配符选择器可以用来设置所有元素的样式，或者设置某个元素下所有元素的样式。通常情况下，通配符选择器会被用在CSS样式文件的开头，用于去除浏览器默认的边框设计，例如下面的代码：

```
*{
margin:0;
padding:0;
}
```

这段代码的意思是所有元素的内边距设置为0个像素，所有元素的外边距设置为0个像素。由于目前可供大家选择的浏览器种类很多，不同的浏览器对于默认边框的距离设置是有差别的，为了能够更好的控制页面的布局效果，设计者往往需要清除浏览器默认的边距设计，这就是CSS样式文件的开头往往都要加上这段代码的原因。

通配符选择器还有一种用法，用于表示某个元素下所有元素的样式。我们都知道，HTML页面中的标签可以嵌套使用，如果想让被嵌套的元素具有相同的样式，就可以使用通配符选择器。例如下面这段代码：

```
.demo *{
color:red;
font-size:24px;
}
```

所有被通配符选择器demo嵌套的元素，都具有相同的颜色和字体大小。

6.4.5 属性选择器

属性选择器可以根据元素的属性及属性值来选择元素。属性选择器的应用非常灵活，我们可以将属性选择器分为以下几种类型。

1. 简单属性选择器

这类属性选择器仅仅关注HTML标签中是否包含匹配的属性，而不关注其属性值是否相

同。例如下面这段代码：

```
*[title]{
color:blue;
}
```

使用通配符选择器和属性选择器，设置所有具有title属性的元素的颜色为蓝色。注意属性选择器应该书写在方括号内。这里的通配符也可以换成特定的HTML元素，例如下面这段代码：

```
a[title]{
color:blue;
}
```

另外，如果需要对多个属性设置相同的样式，还可以将多个属性选择器连接在一起使用，例如下面这段代码：

```
a[href] [title]{
color:blue;
}
```

2. 根据具体属性值选择

根据属性值可以更准确地匹配到HTML元素，这样就可以在具有相同属性的多个元素中缩小选择范围。例如下面这段代码：

```
a[href="http://www.baidu.com"]{
color:red;
}
```

这段代码中仅指定了链接地址为“http://www.baidu.com”的超链接显示为红色，如果HTML页面中还有其他超链接，但是它们的href值不是这个链接地址，那么就不会显示为红色。

与简单的属性选择器类似，根据具体属性值选择的属性选择器也可以将具有多个属性的选择器连接在一起使用。例如下面这段代码：

```
a[href="http://www.baidu.com"][title="百度"]{
color:red;
}
```

这段代码中不仅确切地指定了属性href的值，而且还指定了属性title的值，只有满足这两个条件的超链接元素才可以显示为红色。

3. 根据部分属性值选择

在HTML页面中，很多个相同的HTML元素都设置了同样的属性，它们的属性值相互关联却不完全相同，如果要设置这些元素的属性，就可以使用部分属性值匹配选择的方式。如果需要根据属性值中的某一部分进行选择，则需要使用波浪号（~）。例如下面这段代码：

```
<!doctype html>
<html>
<head>
```

```
<meta charset="utf-8">
<title>6.6</title>
<style type="text/css">
p[class~="sec"]{
color:blue;
}
</style>
</head>
<body>
<p class="sec One">这是第一段内容</p>
<p class="sec Two">这是第二段内容</p>
<p class="sec_Three">这是第三段内容</p>
</body>
</html>
```

运行这段代码后，效果如图6.6所示。

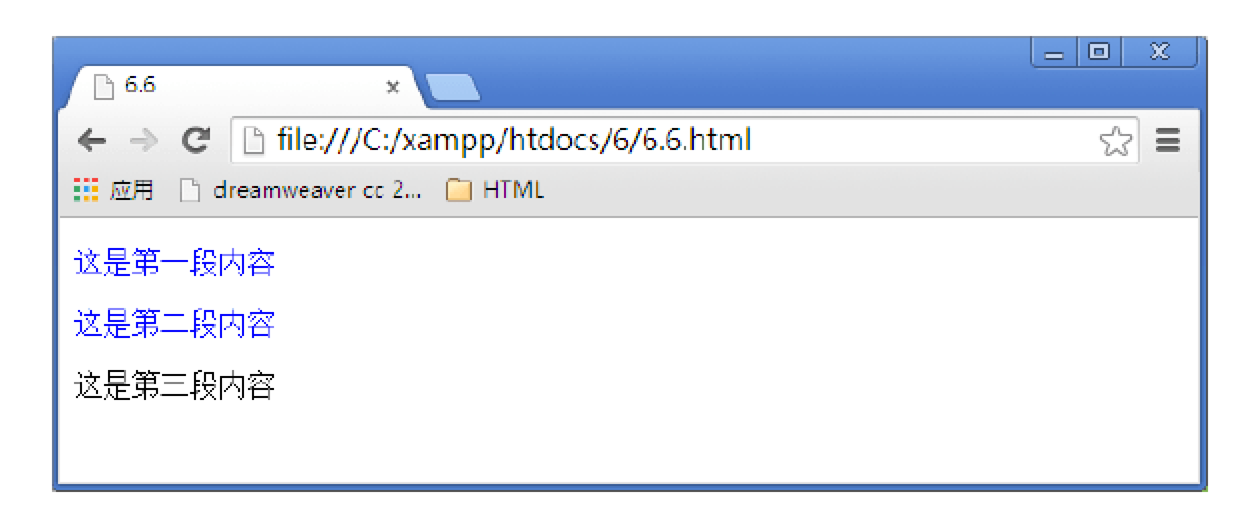


图6.6

可以看到，第一段内容和第二段内容都按照我们的设计应用了样式，但是第三段内容并没有应用样式。这是因为第三段的class属性值中并没有空格，因此并不能被这种部分属性值选择的方式正确匹配。要使用部分属性值选择的属性选择器，必须将HTML元素的属性值设置成具有空格的类型。

4. 子串匹配属性选择器

子串匹配属性选择器是对部分属性值选择器的一个扩充，因为它适用于多种情况下的属性选择器。例如，对以某字符串开头或结束的属性值或者属性值中包含某字符串的属性值进行选择。简单概括如下表所示：

表6.1

类型	描述
[abc^="def"]	选择abc属性值以“def”开头的所有元素
[abc$="def"]	选择abc属性值以“def”结束的所有元素
[abc*="def"]	选择abc属性值中包含子串“def”的所有元素

例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.7</title>
```

```
<style type="text/css">
p[class^="sec"]{
font-size:24px;
}
p[class$="Red"]{
color:red;
}
p[class*="Blue"]{
color:blue;
}
</style>
</head>
<body>
<p class="secRed">这是第一段内容</p>
<p class="secBlue">这是第二段内容</p>
<p class="secYellow">这是第三段内容</p>
</body>
</html>
```

第1个样式匹配以“sec”开头的class属性值，并设置字号为24像素；第2个样式匹配以“Red”结束的class属性值，并设置颜色为红色；第3个样式匹配包含“Blue”字符串的class属性值，并设置样式为蓝色。这里需要注意的是，在匹配字符串与属性值时，应该区分大小写英文字母，否则将无法正确显示样式。运行这段代码后的效果如图6.7所示。



图6.7

6.4.6 嵌套选择器

HTML元素可以嵌套使用，CSS样式也可以通过嵌套来定义选择器。通过嵌套定义的选择器名称多个标记之间用空格分开，例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.8</title>
<style type="text/css">
a{
text-decoration:none;
font-size:24px;
color:red;
}
```

```
</style>
</head>
<body>
<ul>
    <li><a href="http://www.baidu.com">百度</a></li>
    <li><b>Bing</b></li>
</ul>
<a href="http://www.qq.com">腾讯网</a>
</body>
</html>
```

在这段代码中有两个超链接，第1个超链接嵌套在列表的<li>元素中，第2个超链接与列表同级。通过设置超链接的样式，我们去掉了超链接的下划线，设置字号为24个像素，并设置颜色为红色。运行这段代码后，效果如图6.8所示。

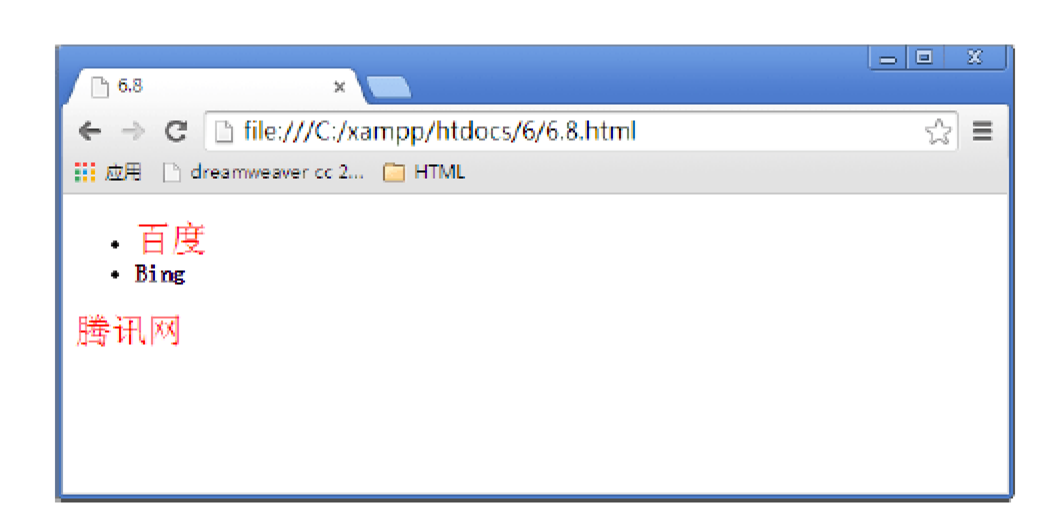


图6.8

我们设置的样式不仅应用到第1个超链接，还应用到第2个超链接。如果希望第1个超链接的颜色显示为绿色，而且字号更小一些，同时又不影响第2个超链接的样式，那么就需要为这两个超链接分别设置id或class属性，并使用id选择器或者class选择器分别设置样式。而使用嵌套选择器可以省略设置id或class属性，CSS样式的代码如下所示：

```
<style type="text/css">
a{
text-decoration:none;
font-size:24px;
color:red;
}
ul li a{
font-size:16px;
color:green;
}
</style>
```

再次刷新页面后，效果如图6.9所示。

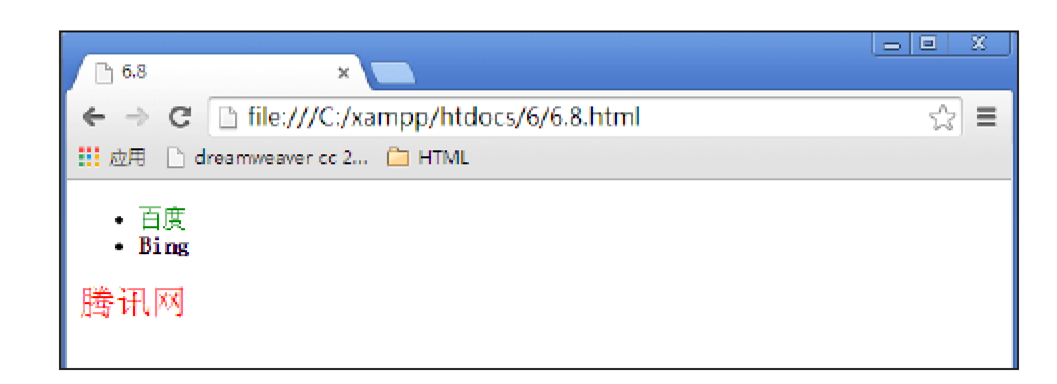


图6.9

6.4.7 链接选择器

链接选择器针对超链接的各种状态设置样式。超链接的状态分为未访问、已访问、鼠标悬浮和点击4种，对应的4种链接选择器如下所示。

（1）link：用于选取未被访问的链接。
（2）visited：用于选取已经访问的链接。
（3）hover：用于选取鼠标悬浮时的链接。
（4）active：用于选取点击时的链接。

例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.9</title>
<style type="text/css">
a:link{
color:blue;
}
a:visited{
color:#A7522B;
}
a:hover{
color:red;
}
a:active{
color: green;
}
</style>
</head>
<body>
<h3>链接选择器</h3>
<a href="http://www.baidu.com" target="_blank">百度</a>
</body>
</html>
```

在这段代码中，a:link选择器设置了超链接没有访问时显示为蓝色，a:visited选择器设置超链接访问后显示为棕色，a:hover选择器设置当鼠标悬浮在超链接上时，超链接显示为红色，a:active选择器设置当点击超链接时显示为黄色。



6.5 CSS内容排版

HTML页面布局和排版是CSS的主要功能。虽然HTML标签都有默认的样式，但是为了丰

富页面的内容，我们仍需要使用CSS重新设置页面的样式。

6.5.1 设置字体

CSS字体属性用于定义文本的字体系列、大小、加粗、风格和变形。CSS中设置了两种类型的字体系列名称，一种是拥有相似外观的字体系列组合，称为通用字体系列，总共有5种，包括Serif字体、Sans-serif字体、Monospace字体、Cursive字体和Fantasy字体；另一种是具体的字体系列，称为特定字体系列，如Times或Courier。

在CSS中使用font-family属性设置字体，例如下面这段代码：

```
h1{
font-family:Cambria
}
```

在这段代码中，我们为标题h1设置字体为Cambria。通常情况下，这样设置是没有问题的，但是如果用户的电脑上没有安装Cambria字体，系统可能会使用通用自体中比较接近的一种字体替代Cambria字体。为了避免出现这种不确定的情况，我们在设置字体的时候可以指定一系列类似的字体，例如下面这段代码：

```
h1{
font-family:Cambria, "Hoefler Text", "Liberation Serif", Times, "Times New Roman", serif
}
```

当系统找不到Cambria字体时，就会继续匹配后面一种字体，以此类推。这里还需要说明的一点是，有些字体的名称可能由多个单词组成，例如Hoefler Text，在设置这种字体时需要使用引号，因为这类字体名称中间有一个空格。某些字体名称可能包含#或者$之类的字符，也需要使用引号。

6.5.2 文字排版

文本是HTML页面中的主要内容，CSS中使用font属性设置文本的样式。font是一系列关于语法和如何使用CSS调整文字的概述。下面我们详细地进行介绍。

（1）font-size：设置文字的大小。

不同大小的文字可以让网页看起来更有层次感。HTML标签也有默认的文字大小，例如HTML标题标签，从<h1>到<h6>文本的大小逐渐减小，而<p>标签表示的段落也有默认的大小。这些标签都不用特别设置文字的大小，除非有特殊的需要。

例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.10</title>
<style type="text/css">
```

```
#defaultTitle{
color:red;
}
#updateTitle{
color:blue;
font-size:20px;
}
</style>
</head>
<body>
<h1 id="defaultTitle">默认标题</h1>
<h1 id="updateTitle">修改后的标题</h1>
</body>
</html>
```

在这段代码中有两个<h1>标签，它们本来应该有相同的文字大小，但是通过CSS样式将第2个<h1>标题文字大小设置为20px，所以运行这段代码后，效果如图6.10所示。

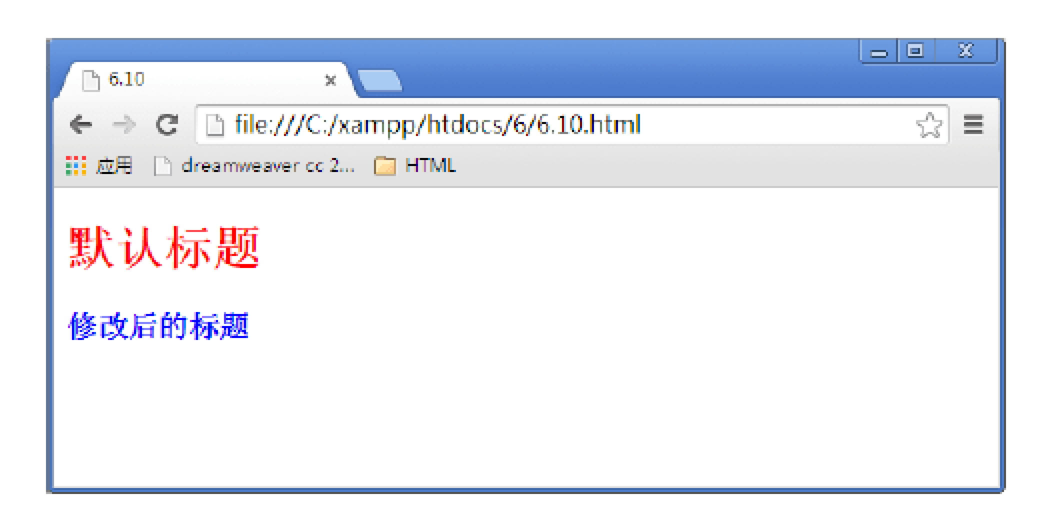


图6.10

（2）font-weight：设置文本的粗细。

在HTML标签中，<b>标签和<strong>标签用于显示粗体的文本。除此之外，我们还可以通过设置font-weight属性控制文本的粗细。在CSS中，从100~900指定了9个级别的文本粗细，100表示最细的文本，900表示最粗的文本。另外还有几种预定义选项，如normal相当于设置字体粗细为400，bold相关于700，bolder相当于比所继承值更粗的一个字体，lighter相当于比继承值更细的一个字体。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.11</title>
<style type="text/css">
#normalStyle{
font-weight:normal;
}
#boldStyle{
font-weight:bold;
}
#bolderStyle{
font-weight:bolder;
```

```
}
#lighterStyle{
font-weight:lighter;
}
</style>
</head>
<body>
<p id="normalStyle">font-weight设置为normal的效果</p>
<p id="boldStyle">font-weight设置为bold的效果</p>
<p id="bolderStyle">font-weight设置为bolder的效果</p>
<p id="lighterStyle">font-weight设置为lighter的效果</p>
</body>
</html>
```

在这段代码中有4个段落标记，每个段落都设置了不同的字体粗细，运行这段代码后，效果如图6.11所示。



图6.11

虽然我们设置了4种不同的字体粗细，但是只能看到正常和加粗两种效果，这是因为大部分浏览器只能显示这两种效果。

（3）text-transform：设置文本的大小写状态。

text-transform属性的值总共有5个，capitalize表示首字母大写，uppercase表示全部大写，lowercase表示全部小写，none表示正常没有变化（默认值），inherit表示继承其父级对象的属性。该属性对中文没有效果。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.12</title>
<style type="text/css">
#capitalizeStyle{
text-transform:capitalize;
}
#uppercaseStyle{
text-transform:uppercase;
}
#lowercaseStyle{
text-transform:lowercase;
}
```

```
#noneStyle{
text-transform:none;
}
#inheritStyle{
text-transform:inherit;
}
</style>
</head>
<body>
<p id="capitalizeStyle">This is a test.首字母大写</p>
<p id="uppercaseStyle">This is a test.全部大写</p>
<p id="lowercaseStyle">This is a test.全部小写</p>
<p id="noneStyle">This is a test.默认效果</p>
<p id="inheritStyle">This is a test.继承效果</p>
</body>
</html>
```

在这段代码中总共有5个段落，分别设置了不同的text-transform属性值，前3种都对应各自的效果，最后两种效果相同，因为最后一个段落应用的样式中，inherit对应的父级元素body并没有设置text-transform属性，所以采用默认的none值，这样它们的效果就一样了。运行这段代码后，效果如图6.12所示。

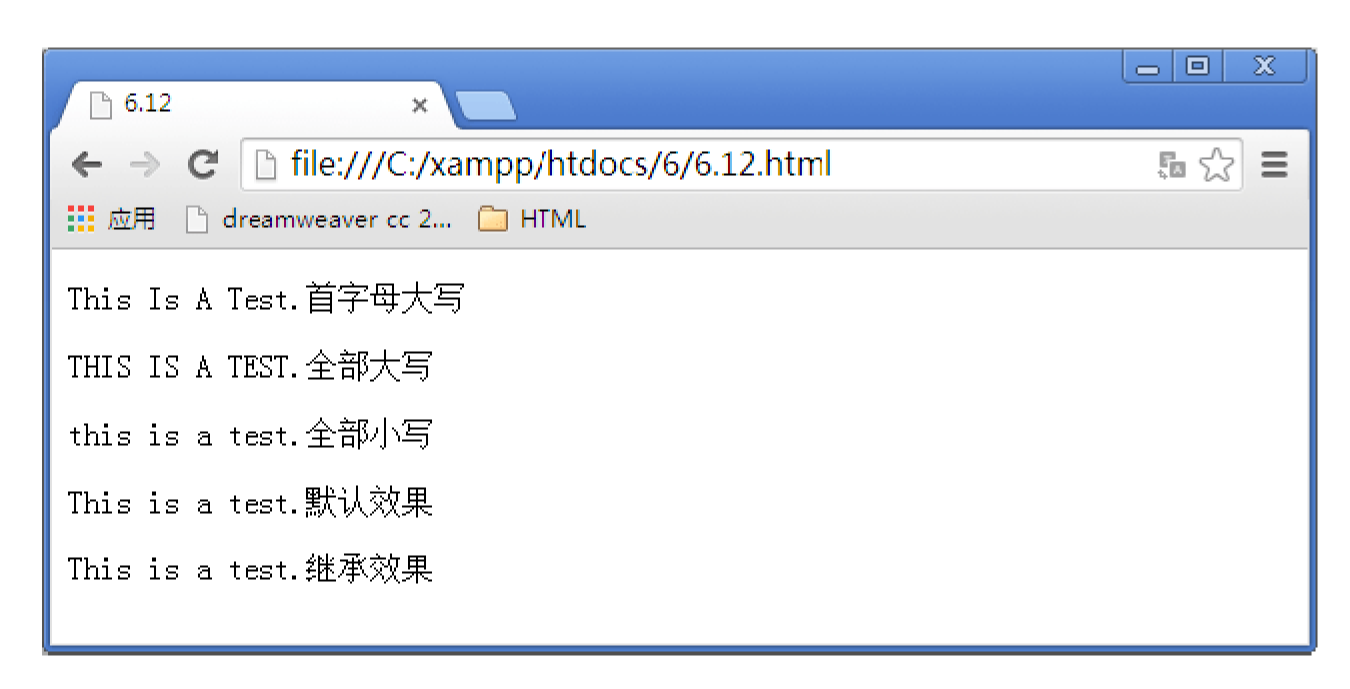


图6.12

（4）text-decoration：设置文本的修饰效果。

CSS中text-decoration的属性值有6个，详细说明如表6.2所示。

表6.2

值	描述
none	定义标准的文本（默认）
underline	定义文本下的一条线
overline	定义文本上的一条线
line-through	定义穿过文本下的一条线
blink	定义闪烁的文本
inherit	规定应该从父元素继承text-decoration属性的值

例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.13</title>
<style type="text/css">
h1{
text-decoration:underline;
}
h2{
text-decoration:overline;
}
h3{
text-decoration:line-through;
}
h4{
text-decoration:blink
}
a{
text-decoration:none;
}
</style>
</head>
<body>
<h1>下划线效果</h1>
<h2>上划线效果</h2>
<h3>横穿线效果</h3>
<h4>闪烁效果</h4>
<a href="http://www.baidu.com" >标准文本效果</a>
</body>
</html>
```

运行这段代码后的效果如图6.13所示。闪烁效果因为使用的很少，目前已经没有浏览器支持这个效果。而HTML中的超链接在默认情况下都有一个下划线的效果，为了让页面文本有一个统一的效果，通常会使用text-decoration:none取消超链接的下划线效果。

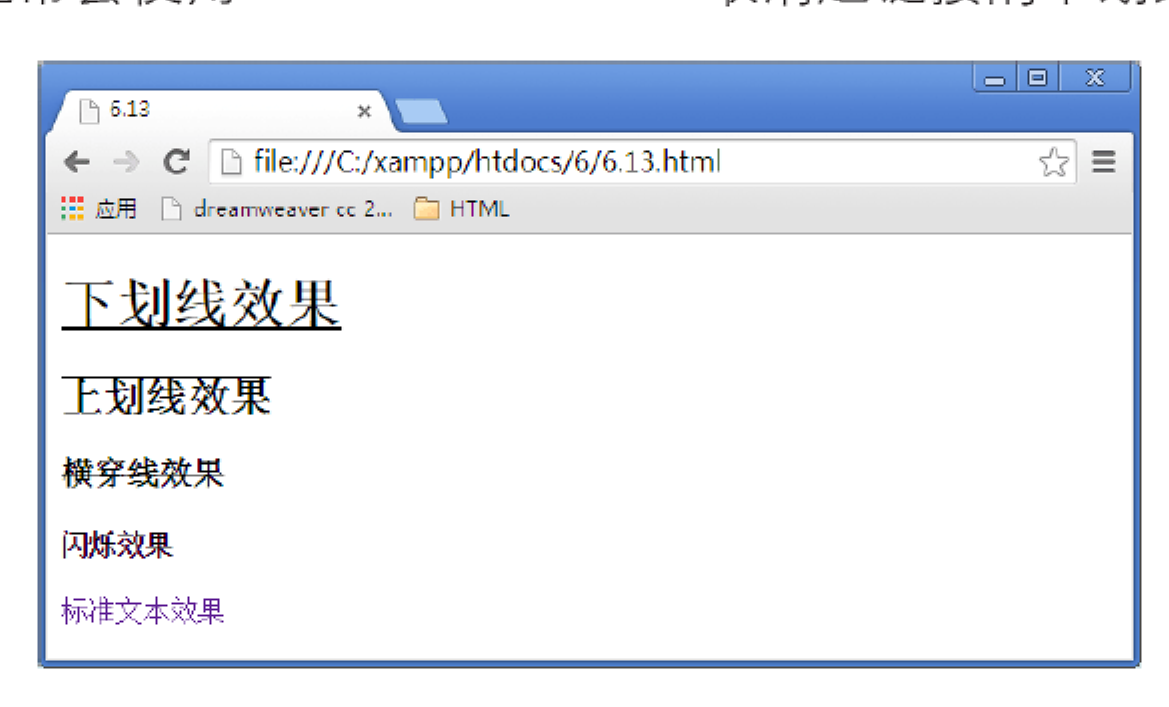


图6.13

（5）font-variant：设置小型的大写字母。

font-variant属性值有3个，normal表示标准的字体，small-caps表示显示小型大写字母的字

体，而inherit表示从父元素继承font-variant属性的值，该属性对中文无效。

例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.14</title>
<style type="text/css">
#normal{
font-variant:normal;
}
#small{
font-variant:small-caps;
}
</style>
</head>
<body>
<h2 id="normal">This is a test.</h2>
<h2 id="small">This is a test.</h2>
</body>
</html>
```

运行这段代码后，效果如图6.14所示。虽然都是<h2>标记的内容，但是第2行文本都是大写字母，除首字母外，其他字母明显小了很多。

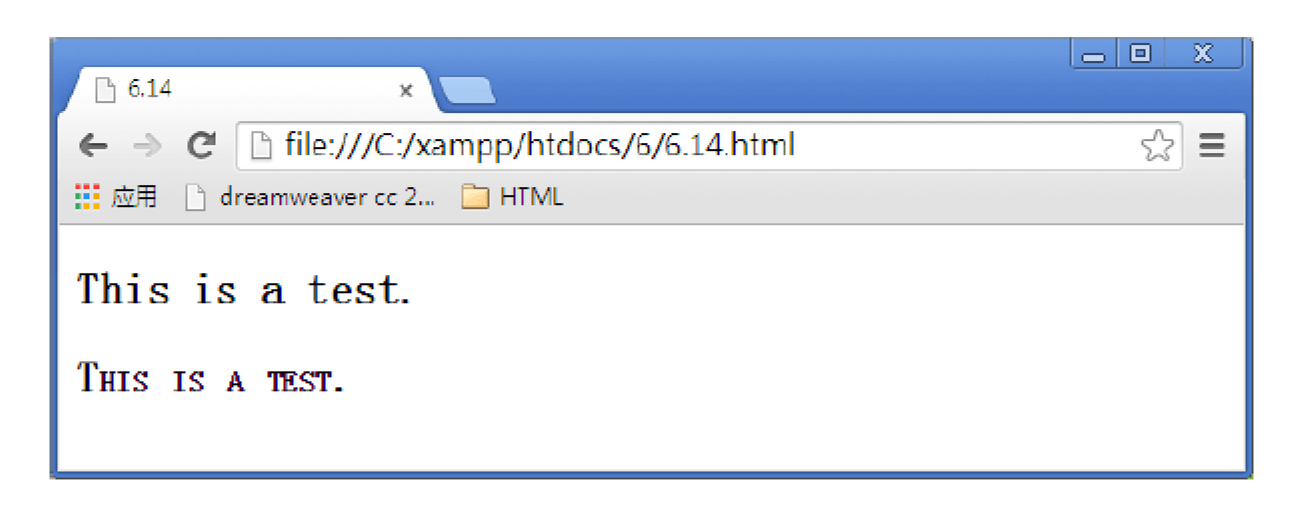


图6.14

（6）letter-spacing和word-spacing：添加空白。

这两个属性都用来添加对应元素的空白。letter-spacing添加字母之间的空白，而word-spacing添加每个单词之间的空白。由于word-spacing修饰的对象是单词，所以对中文无效。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.15</title>
<style type="text/css">
#normal{
letter-spacing:normal;
word-spacing:normal;
}
```

```
    #letterspacing{
    letter-spacing:5px;
    }
    #wordspacing{
    letter-spacing:8px;
    }
    </style>
    </head>
    <body>
    <h2 id="normal">This is a test.</h2>
    <h2 id="letterspacing">This is a test.</h2>
    <h2 id="wordspacing">This is a test.</h2>
    </body>
    </html>
```

运行这段代码后，效果如图6.15所示。

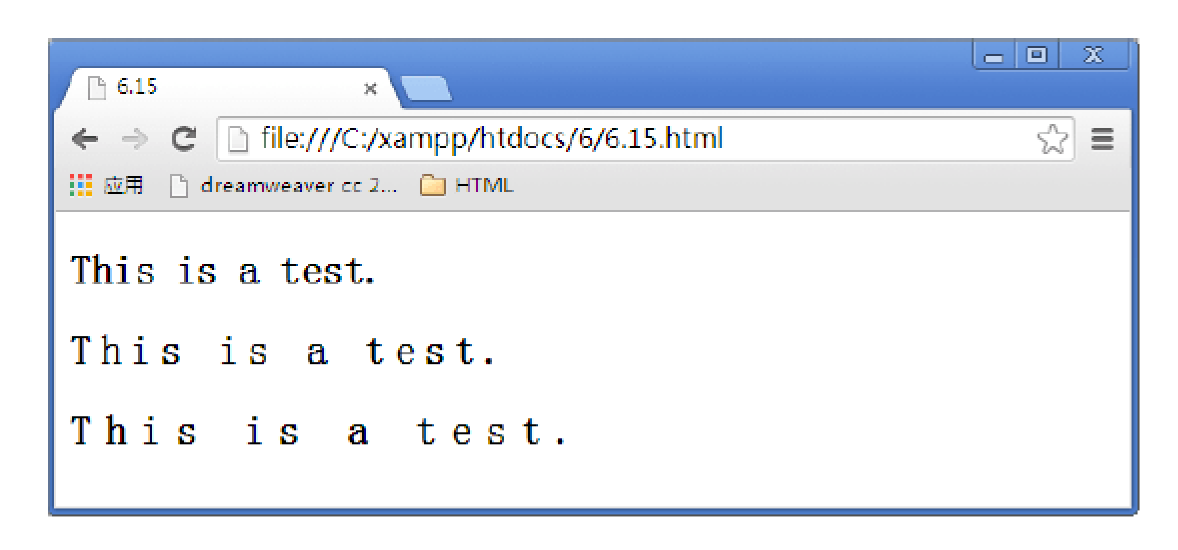


图6.15

（7）font缩写：使用font属性描述以上所有的属性。

以上介绍的这些文字排版属性，都可以使用font属性缩写，这样可以有效提高CSS的书写效率。使用缩写的时候，可以参照以下顺序。

```
    font:font-style font-variant font-weight font-size[/line-height] font-family
```

如果不设置line-height，元素的line-height将会默认为1，而不会从父级元素或body元素继承。使用缩写的时候，某些选项可以省略不写，但是font-size和font-family是必须的。

6.5.3 通栏排版

通栏排版是文字排版中常见的一种排版形式，主要应用于新闻、法律以及文字为主的对象。通栏排版经常使用<p>标签、<h1>到<h6>标签、<span>标签等。将文字作为这些标签的内容，然后再对段落文字应用间距、行距、字号等样式控制，例如下面这段代码：

```
    <!doctype html>
    <html>
    <head>
    <meta charset="utf-8">
    <title>6.16</title>
    <style type="text/css">
    p{
```

```
text-indent:2em;
font-size:16px;
line-height:1.5em;
}
</style>
</head>
<body>
<h1>司法解释</h1>
<p>司法解释，法律解释的一种。属正式解释。司法机关对法律、法规的具体应用问题所做的说明。对某一案件在适用法律上所做的解释，只对该案件有效，没有普遍约束力。最高法院所作的解释，对下级法院通常具有约束力。违背宪法与法律的司法解释无效。</p>
<h2>含义</h2>
<p>司法解释就是依法有权做出的具有普遍司法效力的解释。广义上是指，每一个法官审理每一起案件，都要对法律做出理解，然后才能够具体适用。因此，必须对法律做出解释，才能做出裁判。每一个案件都要这样做。由最高法院对具体适用法律的问题，作出的解释就是司法解释。
中国的司法解释有时特指由最高人民法院和最高人民检察院根据法律赋予的职权，对审判和检察工作中具体应用法律所作的具有普遍司法效力的解释。</p>
<h2>特征</h2>
<p>司法解释只能由有权机关做出。司法解释，具有普遍的司法效力，有关司法机关在办案中应当遵照执行。应该严格依法进行。没有法律具体明确规定的，也要严格依照法律的精神和法律的原则作出解释，供审判工作中具体适用。这就是我们对司法解释的一般理解。</p>
</body>
</html>
```

在这段代码中，我们使用了<h2>和<p>标签通栏排版文字，<h2>标签用于排版标题内容，<p>标签用于排版段落内容，并对<p>标签设置了样式，包括首行缩进、字体大小和行高，运行这段代码后的效果如图6.16所示。



图6.16

6.5.4 分栏排版

分栏排版是指将文字内容排列成两栏或多栏进行显示，常见于报纸的排版形式。在HTML中常用<div>元素来表示某一个区域，如果要分两栏排版，可以使用两个并排显示的<div>元素表示两个分栏，每个<div>元素中的内容就是分栏显示的内容。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.17</title>
<style type="text/css">
p{
text-indent:2em;
font-size:16px;
line-height:1.5em;
}
#layout{
width:600px;
height:400px;
margin:0px auto;
background-color:#C74D4F;
}
.col{
width:250px;
padding:10px 25px;
float:left;
}
</style>
</head>
<body>
<div id="layout">
<div class="col"><h2>含义</h2>
<p>司法解释就是依法有权做出的具有普遍司法效力的解释。广义上是指，每一个法官审理每一起案件，都要对法律做出理解，然后才能够具体适用。因此，必须对法律做出解释，才能做出裁判。每一个案件都要这样做。由最高法院对具体适用法律的问题，作出的解释就是司法解释。中国的司法解释有时特指由最高人民法院和最高人民检察院根据法律赋予的职权，对审判和检察工作中具体应用法律所作的具有普遍司法效力的解释。</p></div>
<div class="col"><h2>特征</h2>
<p>司法解释只能由有权机关做出。司法解释，具有普遍的司法效力，有关司法机关在办案中应当遵照执行。应该严格依法进行。没有法律具体明确规定的，也要严格依照法律的精神和法律的原则作出解释，供审判工作中具体适用。这就是我们对司法解释的一般理解。</p></div>
</div>
</body>
</html>
```

在这段代码中，我们先略去分栏的内容，看一下用<div>元素组织的分栏结构，代码如下所示：

```
<div id="layout">
   <div class="col"></div>
   <div class="col"></div>
</div>
```

这是一个<div>元素内嵌了两个<div>元素，内嵌的两个<div>元素就是分栏的结构。通过id选择器设置了外层<div>元素的宽度和高度，margin:0px auto表示上下外边距为0，左右外边距自动分配，并设置了背景颜色。通过class选择器设置了内层<div>元素的宽度，

padding:10px 25px表示上下内边距为10px，左右内边距为25px。最后一个至关重要的属性float:left表示这两个<div>元素向左浮动，这样就形成了分栏效果。运行这段代码后，效果如图6.17所示。



图6.17

6.5.5 图文混合排版

图像和文本是HTML页面中经常会同时出现的两个内容，处理图像和文本的排版也是CSS的功能之一。例如，我们将图像放在一个<div>元素中，然后插入到页面中，代码如下所示：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.18</title>
<style type="text/css">
p{
text-indent:2em;
font-size:16px;
line-height:1.5em;
}
#layout{
width:600px;
height:400px;
margin:0px auto;
background-color:#C74D4F;
}
</style>
</head>
<body>
<div id="layout">
<h1>司法解释</h1>
<div class="Rimg"><img src="img/img01.jpg" width="40%" height="40%"></div>
<p>司法解释，法律解释的一种。属正式解释。司法机关对法律、法规的具体应用问题所做的说明。对某一
```

```
案件在适用法律上所做的解释，只对该案件有效，没有普遍约束力。最高法院所作的解释，对下级法院通常具有约束力。违背宪法与法律的司法解释无效。</p>
    </div>
    </body>
    </html>
```

这段代码只是对插入的图片大小进行了缩放，其他样式都没有改变，运行这段代码后的效果如图6.18所示。



图6.18

如果要将插入的图片显示在中间位置，可以添加以下样式：

```
.Rimg{
text-align:center;
}
```

刷新页面后的效果如图6.19所示。



图6.19

另外，图文混排效果还可以将文字环绕在图像周围，这就需要添加以下代码：

```
img{
float:left;
padding:10px;
}
```

刷新页面后，效果如图6.20所示。



图6.20

6.5.6 不规则文字环绕

CSS中的float属性可以影响周围元素的排列方式，使用这一特性，我们就可以制作出图像和文字不规则的环绕效果。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.19</title>
<style type="text/css">
*{
margin:0;
padding:0;
color:#972E30;
font-size:14px;
font-weight:bold;
}
#content {
background: url(img/img02.jpg) no-repeat;
}
.blank {
float: left;
clear: left;
margin: 0px 25px 0px 0px;
font-size:9px;
height:16px
}
</style>
</head>
<body>
<div id="content">
```

```
        <div class="blank" style="width:420px;"></div>
        <div class="blank" style="width:406px;"></div>
        <div class="blank" style="width:395px;"></div>
        <div class="blank" style="width:384px;"></div>
        <div class="blank" style="width:375px;"></div>
        <div class="blank" style="width:370px;"></div>
        <div class="blank" style="width:360px;"></div>
        <div class="blank" style="width:364px;"></div>
        <div class="blank" style="width:361px;"></div>
        不<br>
        规<br>
        则<br>
        文<br>
        字<br>
        环<br>
        绕<br>
        效<br>
        果<br>
    </div>
    </body>
    </html>
```

在这段代码中，外层的<div>元素作为一个大容器，内嵌了很多<div>元素和文字。首先在样式中通过通配符选择器设置所有内外边距为0，并设置颜色、字号和加粗效果，然后通过id选择器设置外层<div>的背景图像，最后通过class选择器设置内嵌的<div>元素向左浮动，并及时清理浮动。设置<div>元素的右边距为25px，字号大小为9px，高度为16px，这样设置后，所有内嵌的<div>元素与图像就会浮动起来。最后也是关键的一步，通过行间样式表设置每个内嵌<div>的宽度，这样就可以控制环绕文字距离左边距的距离，达到不规则的效果。运行这段代码后的效果如图6.21所示。

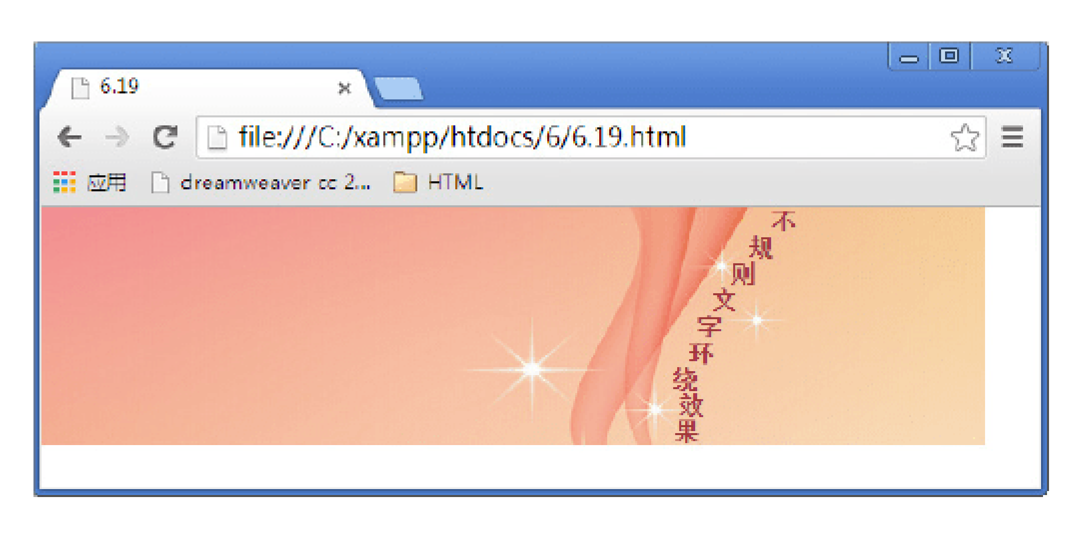


图6.21

6.5.7 全图混排

全图混排通常会应用于相册类的页面中，将很多张图像在一个页面内集中展示，在设置全图混排效果时，我们仍然需要用float这个属性。例如以下HTML页面代码：

```
    <!doctype html>
    <html>
    <head>
```

```
    <meta charset="utf-8">
    <title>6.20</title>
    </head>
    <body>
    <div id="content">
    <div>
           <img src="img/yx01.png">
            <h2>一休</h2>
               <p>本名千菊丸，是日本幕府时期小松天皇的儿子，由于战乱被母亲从小送进安国寺修行，法名一休宗纯。本性善良，乐于助人，而且同样具有小孩子的顽皮和狡猾，在全片中亦是正义的化身。</p>
        </div>
        <div>
            <img src="img/yx02.png">
            <h2>小叶子</h2>
              <p>居住在安国寺附近的小女孩，活泼可爱，心地善良，和爷爷相依为命，经常出入安国寺，对一休有一份真挚特殊的感情。</p>
        </div>
        <div>
            <img src="img/yx03.png">
            <h2>蜷川新佑卫门</h2>
              <p>日本武士兼地方官，武艺高强，剑术超群，同样是正义的代表，严肃起来大义凛然，威严无比，不严肃时又象个小孩子。</p>
        </div>
        <div>
            <img src="img/yx04.png">
            <h2>足利义满将军</h2>
              <p>就是个大孩子，性情喜怒无常，自身养尊处优、四体不勤、五谷不分，而且经常提很多无理要求为难下属和百姓。</p>
        </div>
        <div>
            <img src="img/yx05.png">
            <h2>弥生小姐</h2>
              <p>桔梗店老板的女儿，有钱人家的大小姐，为人傲慢、骄横，和父亲一起总是以各种方式刁难一休，遇到困难又总是来哀求一休帮忙。最大的乐趣就是看到一休出丑。</p>
        </div>
        <div>
            <img src="img/yx06.png">
            <h2>长老</h2>
            <p>本名外鉴，是安国寺的主持，德高望重，慈悲为怀，对弟子要求十分严厉。</p>
        </div>
    </div>
    </body>
    </html>
```

在这段代码中，最外边的<div>元素作为主容器，内嵌的<div>元素中又分别内嵌了图像<img>、标题<h2>和段落<p>元素，没有设置任何样式的效果如图6.22所示。



图6.22

首先为内嵌的<div>元素设置样式，相关代码如下：

```
#content div{
border:solid 1px #37662C;
float:left;
margin:5px;
padding:3px;
width:400px;
height:300px;
text-align:center;
overflow:hidden;
}
```

在这段代码中，首先设置了<div>元素的边框，使用float属性让<div>元素向左浮动，然后设置内外边距，以及<div>元素的宽度和高度，使用text-align:center属性设置内容居中显示，并使用overflow:hidden属性设置隐藏超出的内容。刷新页面后效果如图6.23所示。



图6.23

下面设置<div>元素中的图像样式，相关代码如下，刷新页面后的效果如图6.24所示。

```
#content div img{
```

```
width:260px;
height:180px;
}
```



图6.24

最后设置<div>元素中段落的样式，相关代码如下：

```
#content div p{
font-size:10px;
text-indent:2em;
text-align:left;
}
```

这里设置段落中的字号为10px，首行缩进两个字符，并让段落文本居左对齐，刷新页面后的效果如图6.25所示。



图6.25

6.5.8 表格和边框

如今大家都习惯了使用DIV+CSS布局页面，其实Table+CSS曾经也是流行的页面布局方式。下面我们详细介绍CSS样式如何美化表格和边框的外观。

（1）表格边框：使用border属性可以设置表格的边框。例如下面这段代码，使用border属性设置了<table>、<th>和<td>的边框属性，效果如图6.26所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.21</title>
<style type="text/css">
table,th,td{
border:solid 1px #3F0C0D;
}
</style>
</head>
<body>
<table>
<tr>
        <th>表头</th>
        <th>表头</th>
        <th>表头</th>
    </tr>
    <tr>
        <td>表格的内容</td>
        <td>表格的内容</td>
        <td>表格的内容</td>
    </tr>
</table>
</body>
</html>
```

（2）折叠边框：使用border-collapse属性设置是否将表格边框折叠为单一边框。如图6.26所示，虽然给表格添加了边框，但是显示的是多个边框，其中包括了表格边框、表头边框和单元格边框。如果为表格添加下面的样式，就可以将表格的边框显示为单一的边框，效果如图6.27所示。

```
table{
border-collapse:collapse;
}
```

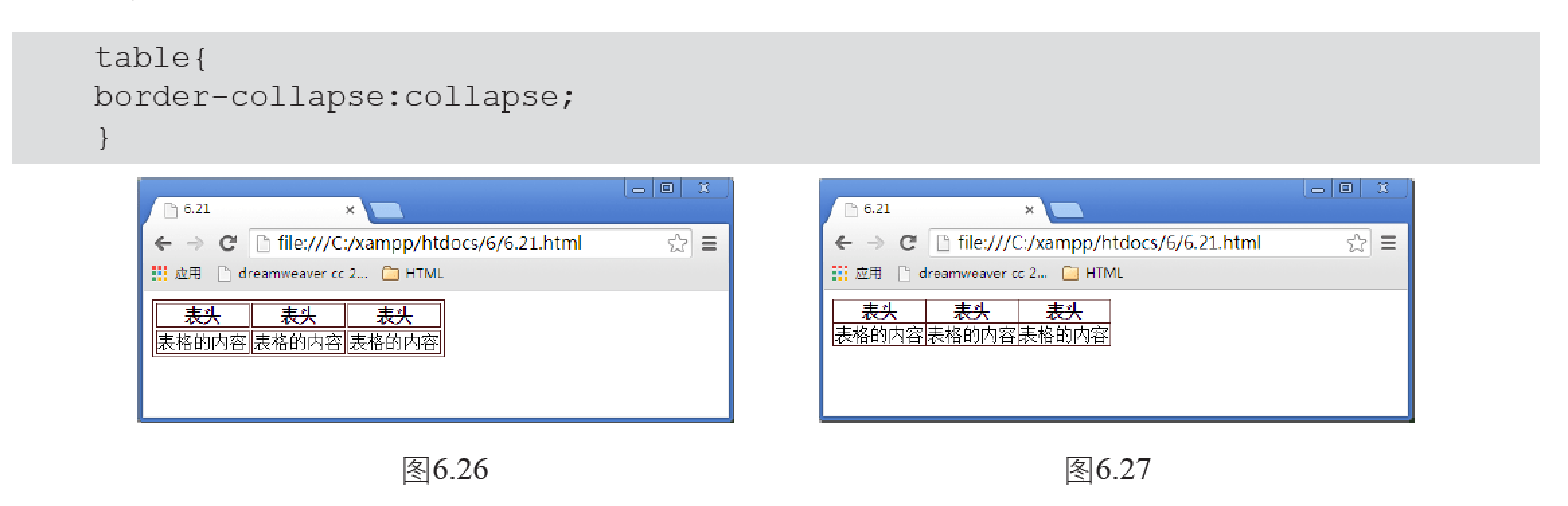


图6.26　　　　图6.27

（3）表格的宽度和高度：通过width和height属性设置表格的宽度和高度。

（4）表格文本对齐：使用text-align和vertical-align属性设置表格中文本的对齐方式。使用text-align属性设置水平对齐方式，如左对齐、右对齐或者居中；使用vertical-align属性设置垂直对齐方式，如顶部对齐、底部对齐或居中对齐。

（5）表格内边距：使用padding属性可以设置表格内边距。例如下面这段代码，表头内

边距为5px，单元格内边距为10px，效果如图6.28所示。

```
th{
padding:5px;
}
td{
padding:10px;
}
```

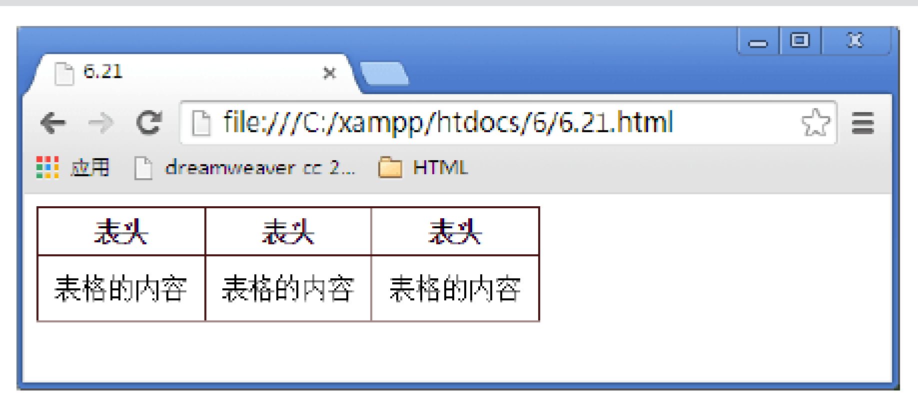


图6.28

（6）表格的颜色：表格中边框的颜色通过border属性进行设置，表格中文本的颜色通过color属性进行设置，表格中背景的颜色通过background-color属性进行设置。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>6.21</title>
<style type="text/css">
table{
border-collapse:collapse;
}
table,th,td{
border:solid 1px #3F0C0D;
}
th{
padding:5px;
background-color:#9F761C;
color:white;
}
td{
padding:10px;
background-color:#C2B6D4;
color:#233673;
}
</style>
</head>
<body>
<table>
<tr>
    <th>表头</th>
        <th>表头</th>
        <th>表头</th>
    </tr>
    <tr>
    <td>表格的内容</td>
        <td>表格的内容</td>
```

```
            <td>表格的内容</td>
        </tr>
    </table>
    </body>
    </html>
```

在这段代码中，分别对表头的背景色、单元格的背景色、表头字体颜色和单元格字体颜色，以及边框颜色进行了设置。运行后的效果如图6.29所示。

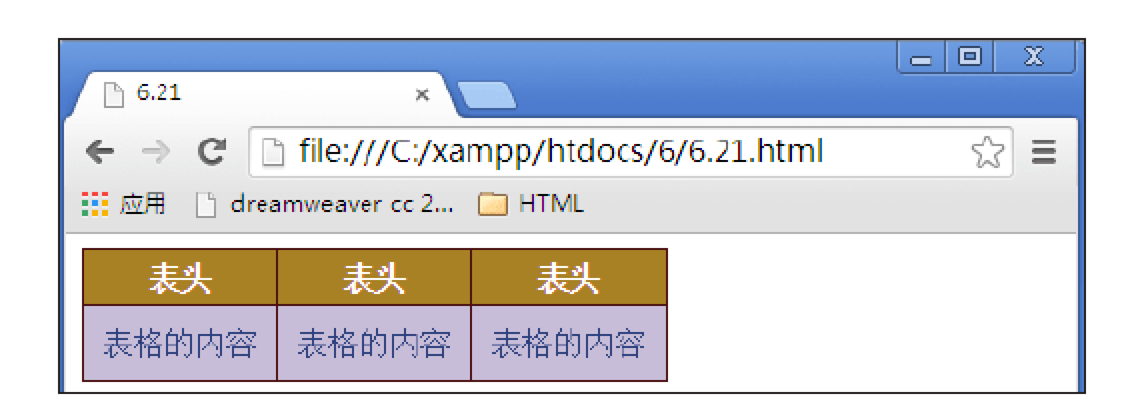


图6.29

（7）边框样式：样式是边框最重要的一个方面，因为如果不设置边框样式，将无法显示边框。在CSS中可以通过border-style属性设置10种不同的边框样式，这10种边框样式如表6.3所示。

表6.3

值	描述
none	定义无边框
hidden	与“none”相同。不过应用于表时除外，对于表，hidden用于解决边框冲突
dotted	定义点状边框。在大多数浏览器中呈现为实线
dashed	定义虚线。在大多数浏览器中呈现为实线
solid	定义实线
double	定义双线。双线的宽度等于border-width的值
groove	定义3D凹槽边框。其效果取决于border-color的值
ridge	定义3D垄状边框。其效果取决于border-color的值
inset	定义3D inset边框。其效果取决于border-color的值
outset	定义3D outset边框。其效果取决于border-color的值
inherit	规定应该从父元素继承边框样式

如图6.30所示为部分边框样式效果。



图6.30



6.6 制作预览幻灯片

本节将使用前面学习到的知识，使用CSS制作一个具有预览功能的幻灯片，效果如图6.31所示。将鼠标悬停在左边或者右边的图片上时，原来显示一半的图片就会全部显示出来，当点击鼠标时，可以跳转到下一个图片。详细的制作步骤如下：

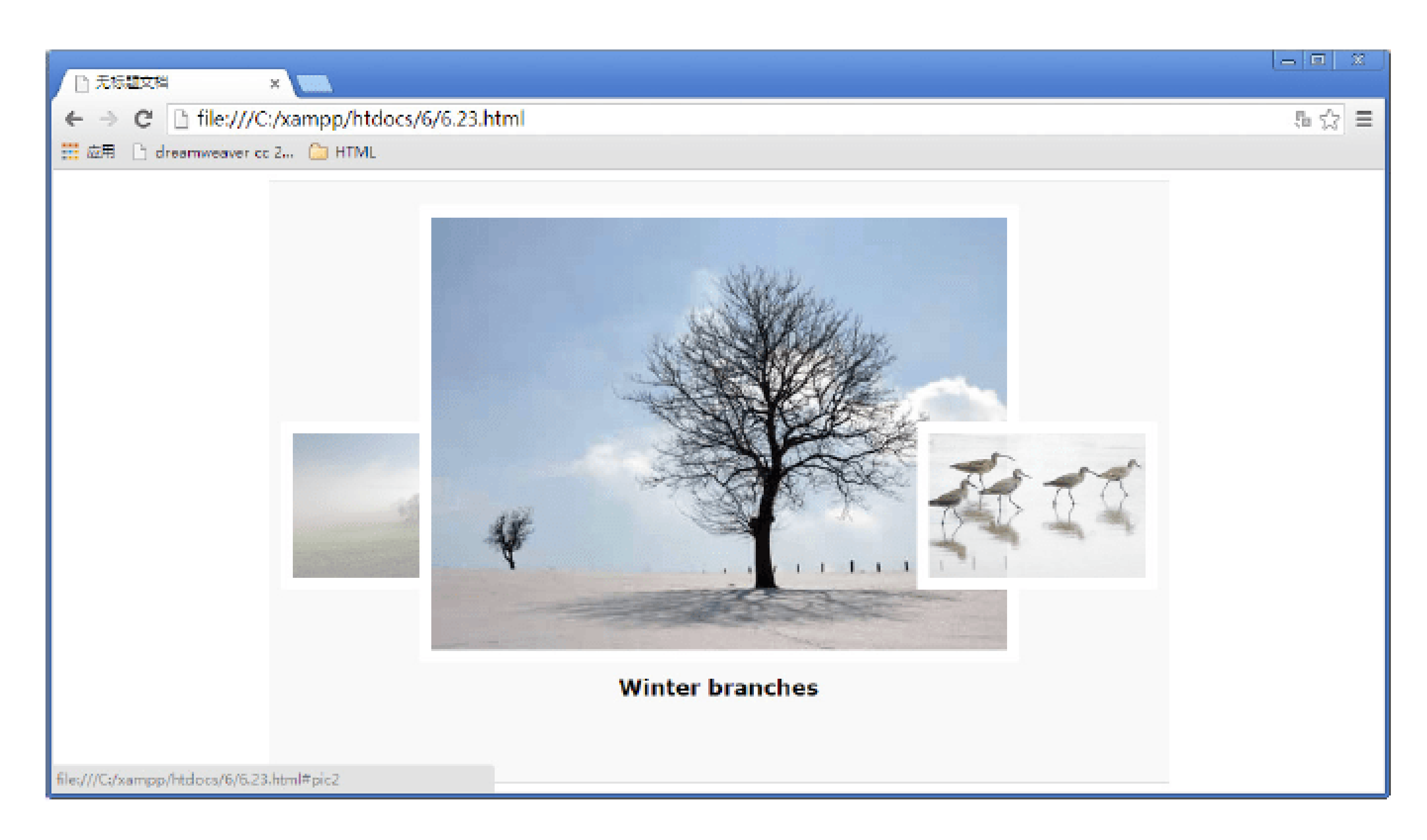


图6.31

01 首先准备10张尺寸一样的图片，分别为它们编号从pic1到pic10。

02 新建一个HTML页面，组织页面结构，相关代码如下所示：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>无标题文档</title>
</head>
<body>
    <div id="gallery">
        <div id="fullsize">
            <div id="pic1">
                <img src="img/pic1.jpg" alt="Winter branches" />
                <a class="previous" href="#pic10">
                <img src="img/pic10.jpg" alt="Trees in the mist" />
                </a>
                <a class="next" href="#pic2">
                <img src="img/pic2.jpg" alt="Wading birds" />
                </a>
                <h3>Winter branches</h3>
            </div>
        </div>
    </div>
</body>
</html>
```

在这段代码中，使用了3层嵌套的div，最外边一层div用于设置幻灯片场景，中间的一层div用于切换幻灯片时固定位置，最内层的div用于显示图片和标题。最内层的3张图片中，第1张图片是当前显示的幻灯片，第2张图片是前一次显示的幻灯片，第3张图片是下一次显示的幻灯片。这里使用锚点切换每组图片显示的位置，后面还会在第3次div的下面依次添加其他各组图片。这段代码在浏览器中的显示效果如图6.32所示。



图6.32

03 首先设置最外层div的样式，分别指定它的宽度和高度；设置margin属性上下为0，左右自适应，可以将图片左右居中显示；使用font-family属性设置页面字体；使用background属性设置背景色；最后设置div的上下边框效果。相关代码如下所示：

```
#gallery {
width: 750px;
height: 500px;
margin: 0 auto;
font-family: verdana, arial, sans-serif;
background: #f8f8f8;
border-top: 1px solid #ddd;
border-bottom: 1px solid #ddd;
}
```

刷新浏览器后的效果如图6.33所示。

04 通过以上设置，可以看到原本横向排列的3张图片，由于设置了div容器的尺寸，无法横向排列后自动变为竖向排列，且3张图片超出了背景区域。为了让图片局限在背景区域中显示，我们需要将超出边框的图片隐藏起来，相关设置如下所示：

```
#gallery #fullsize {
height: 500px;
width: 750px;
overflow: hidden;
text-align: center;
}
```

由于外层div设置了相对定位，且左右方向上居中显示，所以在这段代码中，首先设置该div与外层div的尺寸相同，然后设置overflow属性为hidden，将超出div边框的内容隐藏起来，最后设置text-align属性为center，将图片在这个div中居中显示。刷新浏览器后的效果如图6.34所示。

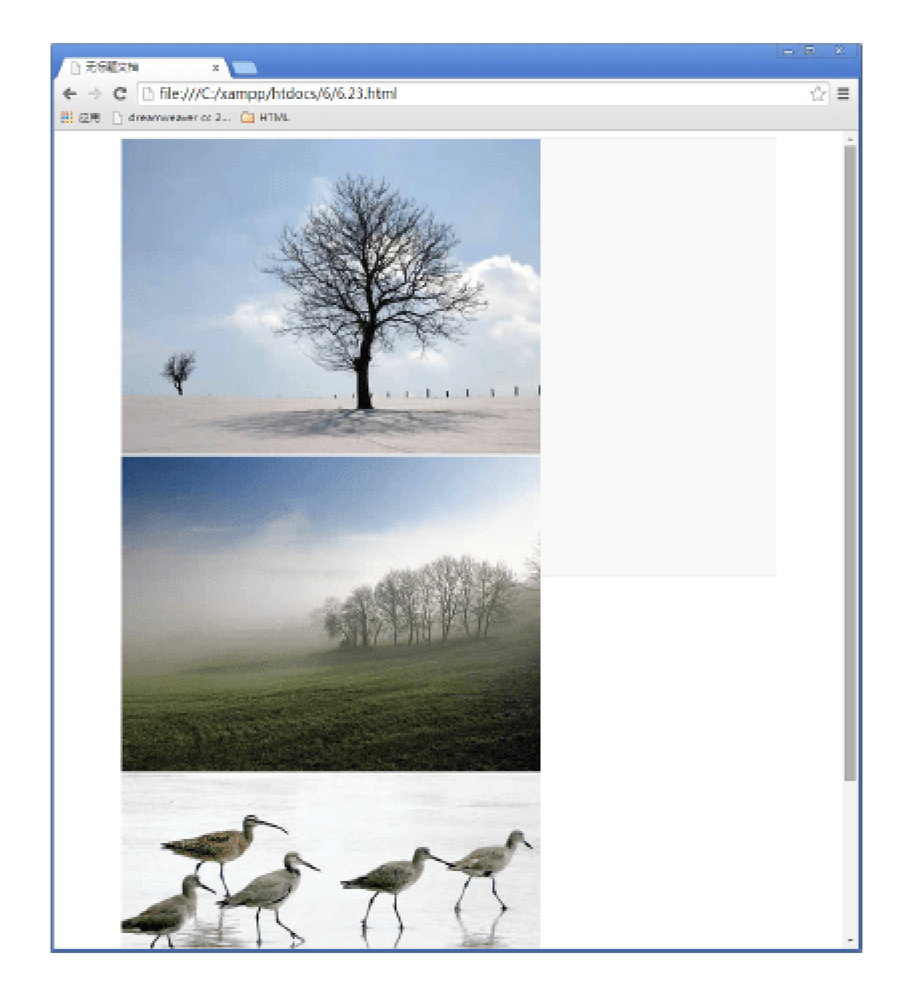


图6.33

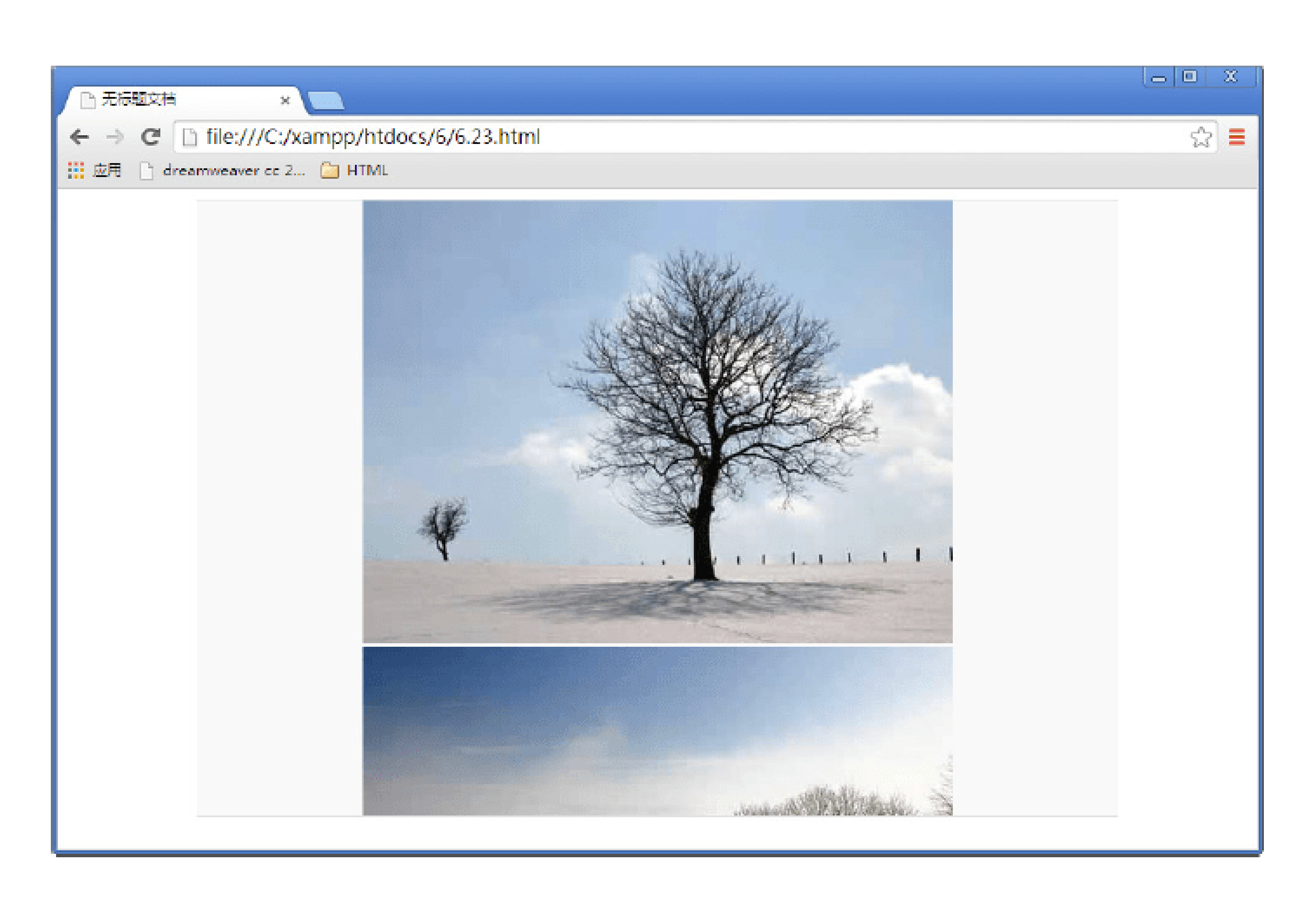


图6.34

05 最内层的div，我们需要设置其宽度与外边的div相同，高度可以适当的高出一些，因为超出边框的部分会被隐藏。为了让幻灯片显示的更美观一些，需要给图片设置一定的内边距，这里设置为20个像素，最后设置position属性为relative。相关代码如下所示：

```
#gallery #fullsize div {
width: 750px;
height: 700px;
padding-top: 20px;
position: relative;
}
```

刷新页面后的效果如图6.35所示。

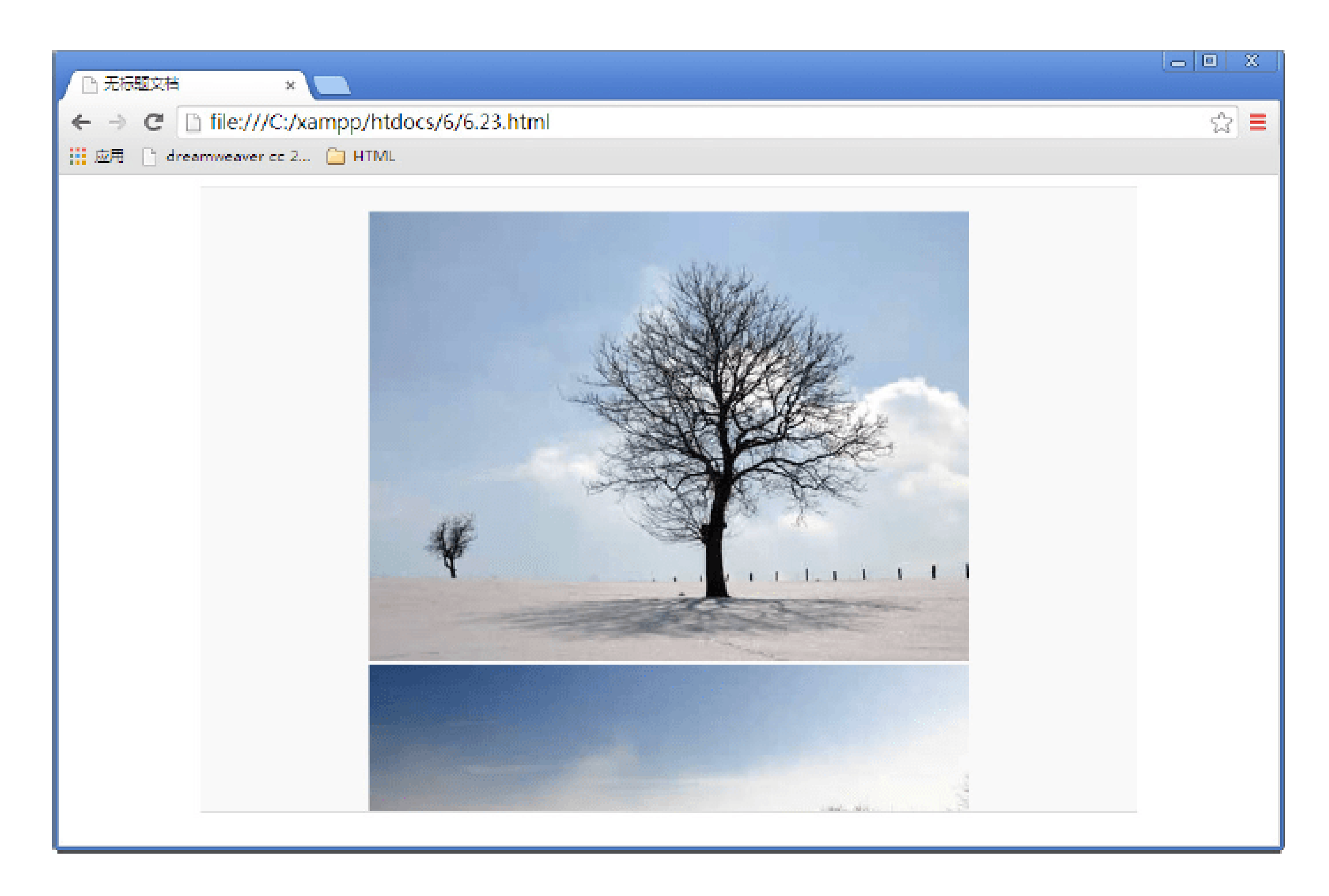


图6.35

06 为div中的图片设置一个10像素的边框，并设置颜色；设置position属性为relative；最后设置z-index属性为500。此处设置元素的堆叠属性，是为了后面凸显小图片效果做铺垫。相关代码如下所示：

```
#gallery #fullsize div img {
border: 10px solid #fff;
position: relative;
z-index: 500;
}
```

刷新浏览器后的效果如图6.36所示，该效果与上图的主要区别在于图片的边框。

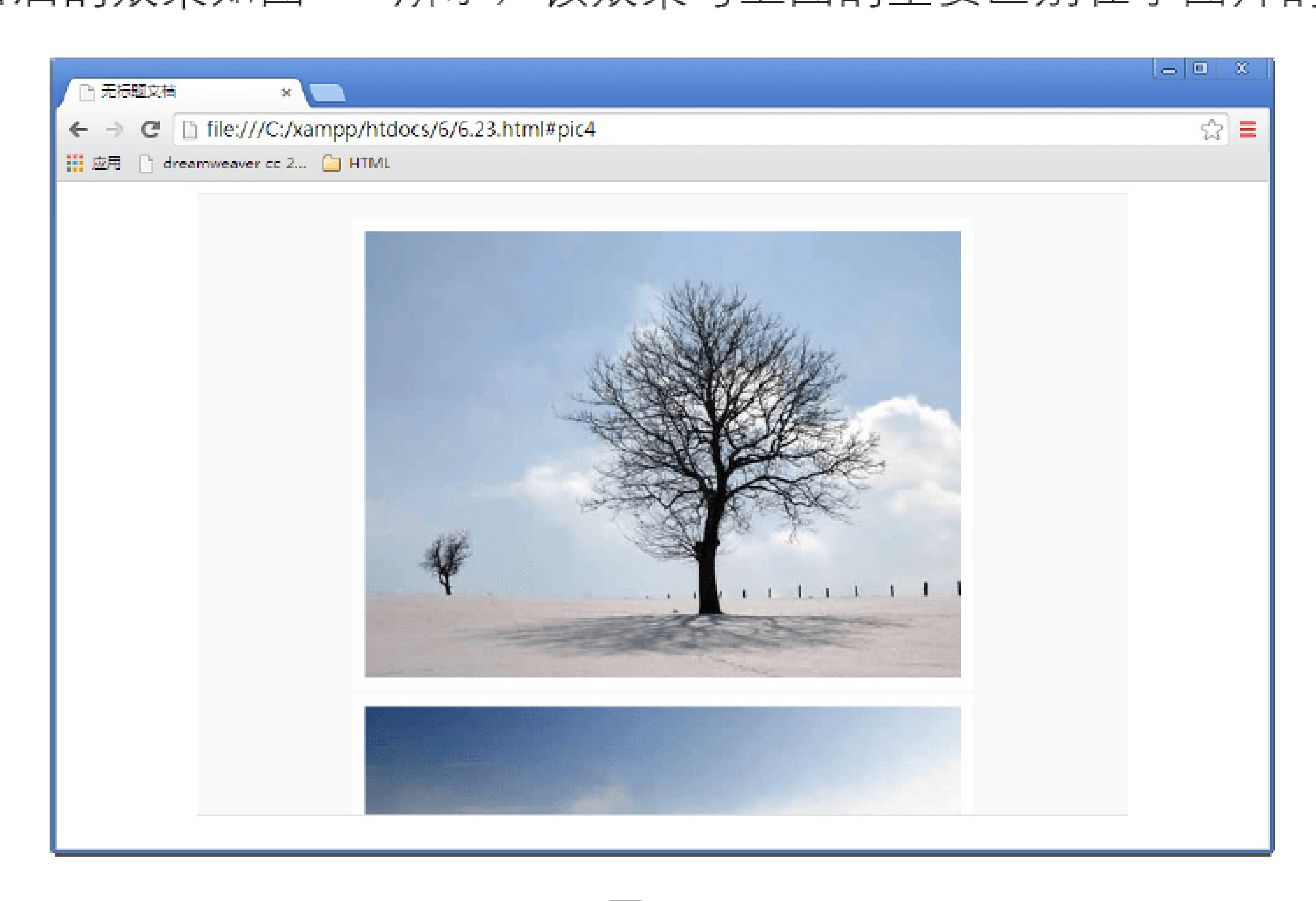


图6.36

07 目前这3张图片还是垂直排列在一起，并且超出外层div的图片都被隐藏起来。为了能让后两张图片分别显示在第1张图片的左右两边，这里需要将position属性设置为absolute，在绝对定位的作用下，分别设置超链接的left和right属性为10，然后设置top属性为200，更改z-index属性为10，这样这两张图片就与第1张图片堆叠显示，并置于第1张图片的后面，相关

代码如下所示：

```
#gallery #fullsize a.previous {
position: absolute;
left: 10px;
top: 200px;
z-index: 10;
}
#gallery #fullsize a.next {
position: absolute;
right: 10px;
top: 200px;
z-index: 10;
}
```

刷新浏览器后的效果如图6.37所示。

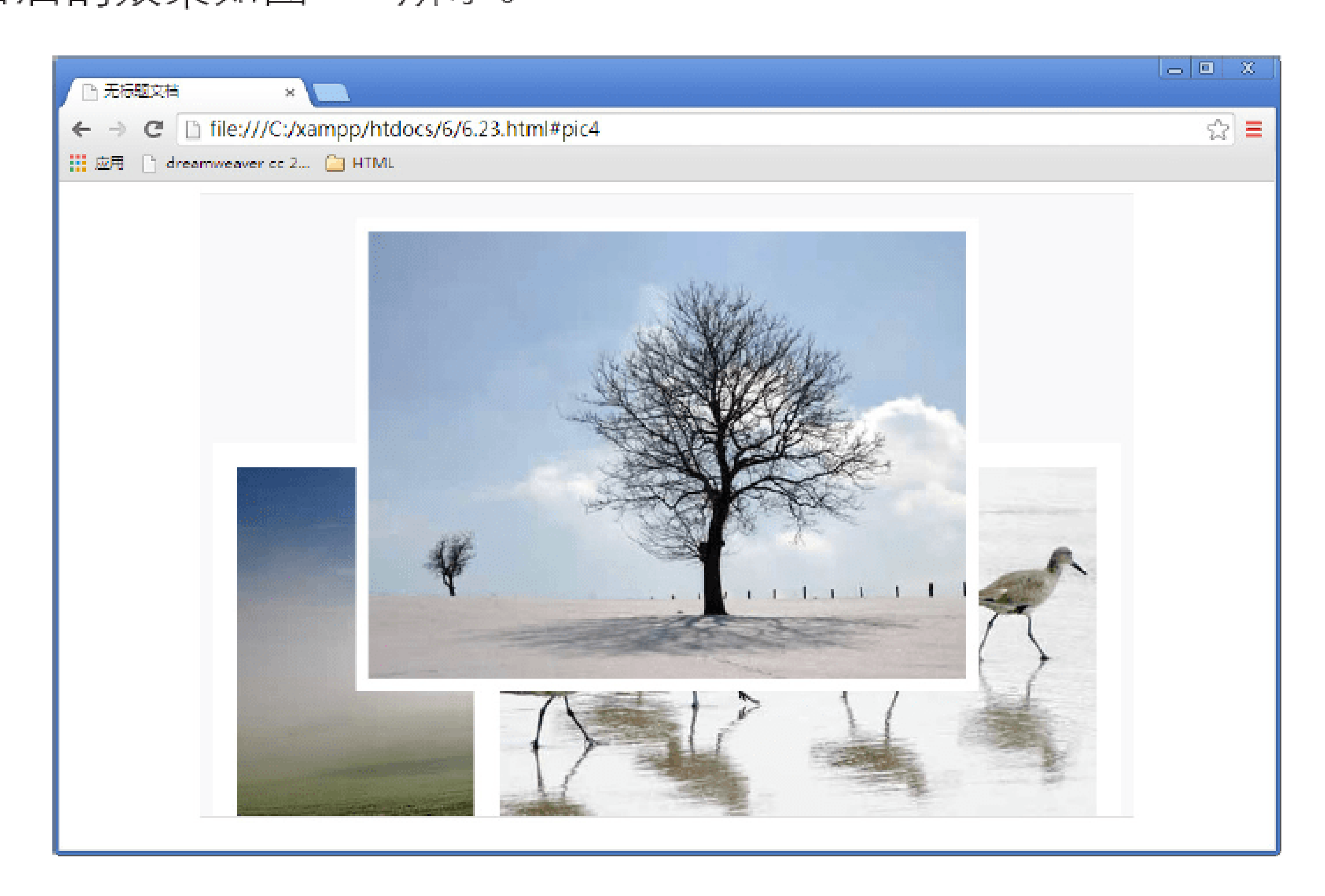


图6.37

08 接下来需要重新设置左右两张图片的尺寸，并为图片设置一个滤镜效果，相关代码如下：

```
#gallery #fullsize a.previous img, #gallery #fullsize a.next img {
width: 180px;
height: 120px;
filter: alpha(opacity=40);
opacity: 0.4;
}
```

刷新浏览器后的效果如图6.38所示。

09 整个幻灯片的结构已经完成，接下来需要为其添加一些动画效果。当鼠标悬停在超链接上时，更改超链接的z-index属性，让图片呈现在幻灯片的前面，并更改图片的滤镜效果，相关代码如下所示：

```
#gallery #fullsize a:hover {
z-index: 600;
```

```
}
#gallery #fullsize a:hover img {
filter: alpha(opacity=80);
opacity: 0.8;
}
```

刷新浏览器后，将鼠标悬停到小图片上的效果如图6.39所示。

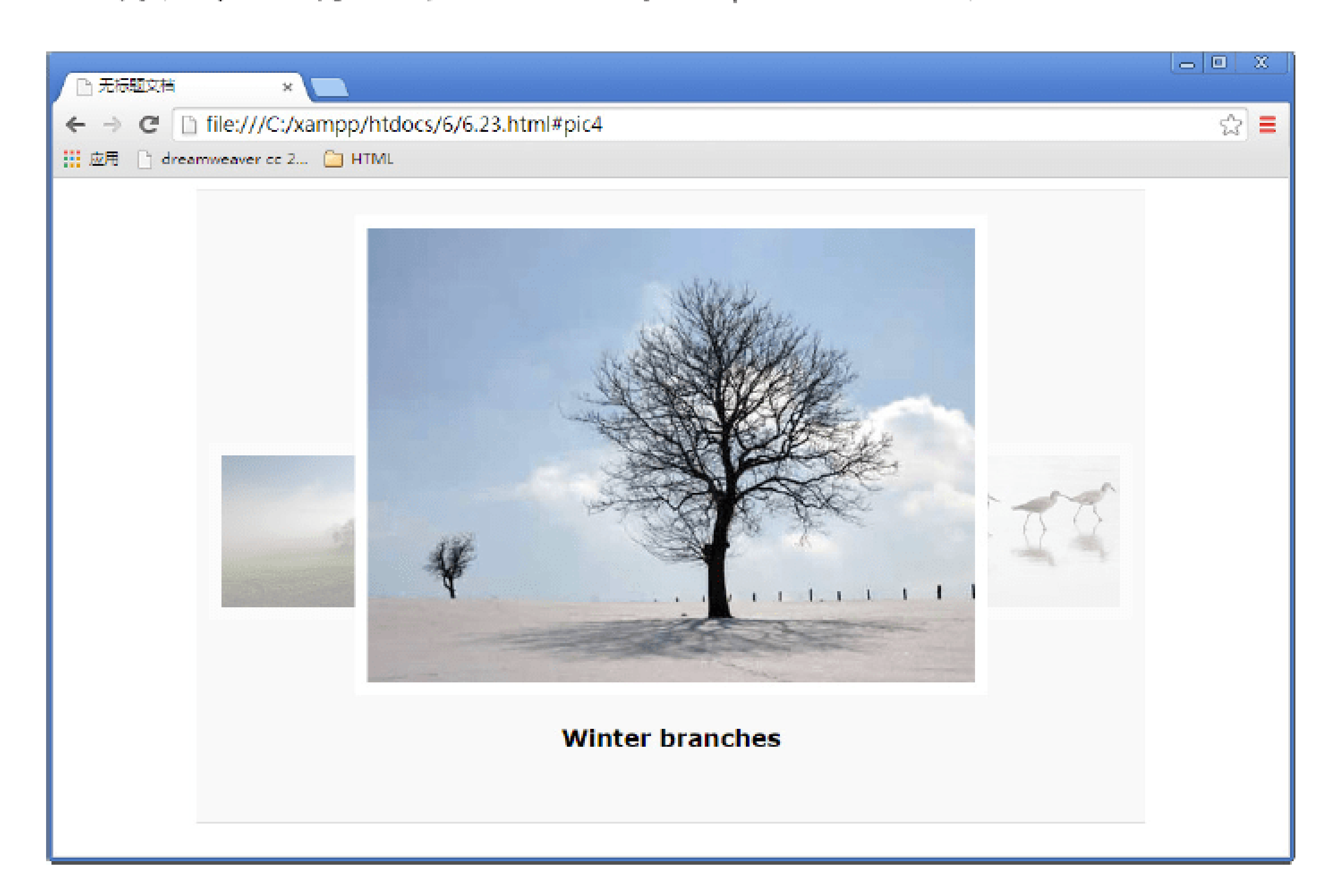


图6.38

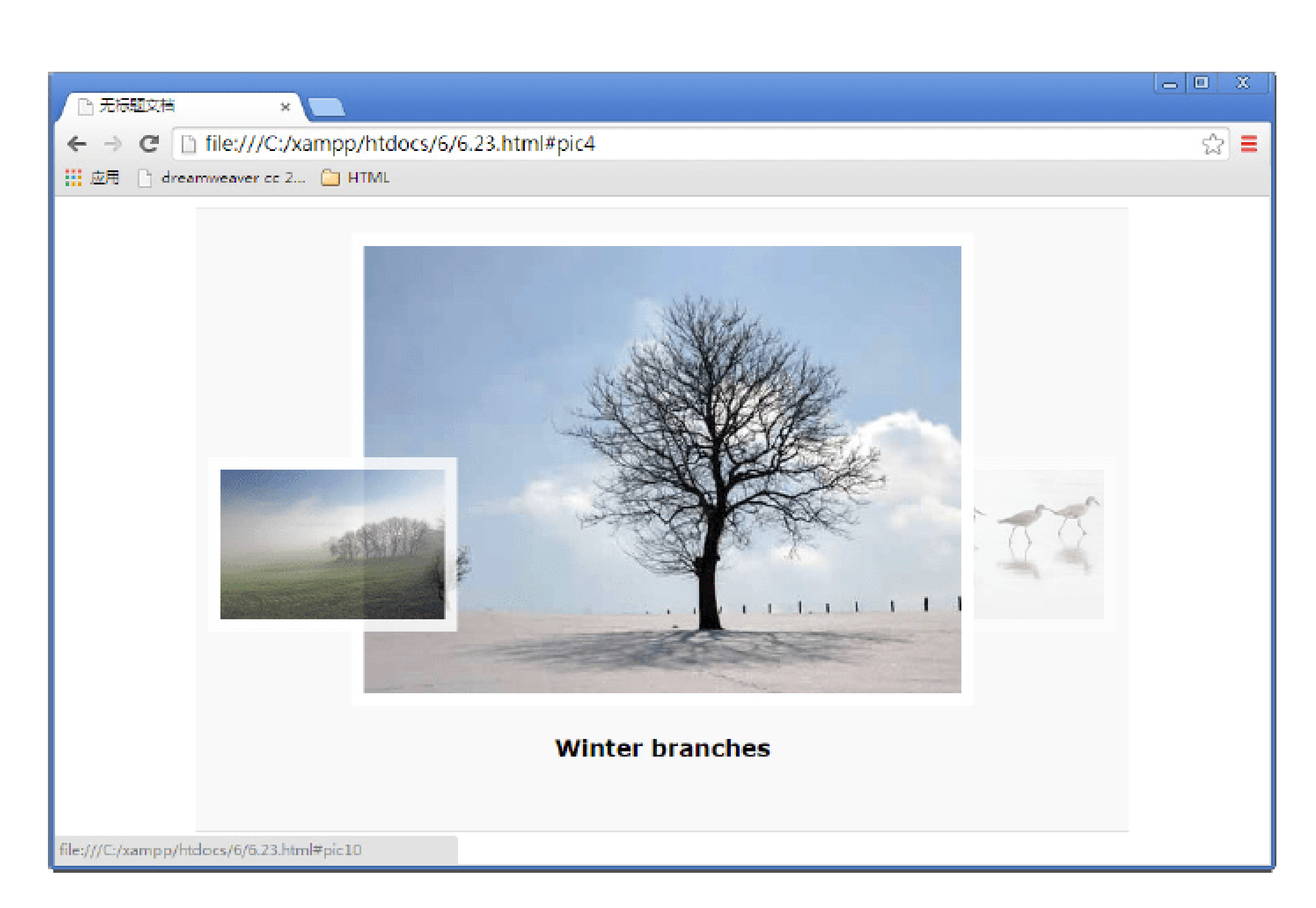


图6.39

10 最后，为页面添加其他的div，当点击超链接的图片时，通过锚点定位到指定的div，这样就形成了有预览功能的幻灯片效果，完整的HTML代码如下所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>无标题文档</title>
<style>
```

```
#gallery {
width: 750px;
height: 500px;
margin: 0 auto;
font-family: verdana, arial, sans-serif;
background: #f8f8f8;
border-top: 1px solid #ddd;
border-bottom: 1px solid #ddd;
}
#gallery #fullsize {
height: 500px;
width: 750px;
overflow: hidden;
text-align: center;
}
#gallery #fullsize div {
width: 750px;
height: 700px;
padding-top: 20px;
position: relative;
}
#gallery #fullsize div img {
border: 10px solid #fff;
position: relative;
z-index: 500;
}
#gallery #fullsize a.previous {
position: absolute;
left: 10px;
top: 200px;
z-index: 10;
}
#gallery #fullsize a.next {
position: absolute;
right: 10px;
top: 200px;
z-index: 10;
}
#gallery #fullsize a.previous img, #gallery #fullsize a.next img {
width: 180px;
height: 120px;
filter: alpha(opacity=40);
opacity: 0.4;
}
#gallery #fullsize a:hover {
z-index: 600;
}
#gallery #fullsize a:hover img {
filter: alpha(opacity=80);
opacity: 0.8;
}
```

```
</style>
</head>
<body>
<div id="gallery">
  <div id="fullsize">
  <div id="pic1">
              <img src="img/pic1.jpg" alt="Winter branches" />
              <a class="previous" href="#pic10">
              <img src="img/pic10.jpg" alt="Trees in the mist" />
              </a>
              <a class="next" href="#pic2">
              <img src="img/pic2.jpg" alt="Wading birds" />
              </a>
              <h3>Winter branches</h3>
          </div>
          <div id="pic2">
              <img src="img/pic2.jpg" alt="Wading birds" />
              <a class="previous" href="#pic1">
                  <img src="img/pic1.jpg" alt="Winter branches" />
              </a>
              <a class="next" href="#pic3">
                  <img src="img/pic3.jpg" alt="Bird on a post" />
              </a>
              <h3>Wading birds</h3>
          </div>
          <div id="pic3">
              <img src="img/pic3.jpg" alt="Bird on a post" />
              <a class="previous" href="#pic2">
                  <img src="img/pic2.jpg" alt="Wading birds" />
              </a>
              <a class="next" href="#pic4">
                  <img src="img/pic4.jpg" alt="Early bloomers" />
              </a>
              <h3>Bird on a post</h3>
          </div>
          <div id="pic4">
              <img src="img/pic4.jpg" alt="Early bloomers" />
              <a class="previous" href="#pic3">
                  <img src="img/pic3.jpg" alt="Bird on a post" />
              </a>
              <a class="next" href="#pic5">
                  <img src="img/pic5.jpg" alt="Green lizard" />
              </a>
              <h3>Early bloomers</h3>
          </div>
          <div id="pic5">
              <img src="img/pic5.jpg" alt="Green lizard" />
              <a class="previous" href="#pic4">
                  <img src="img/pic4.jpg" alt="Early bloomers" />
              </a>
              <a class="next" href="#pic6">
```

```
                <img src="img/pic6.jpg" alt="Feeding the birds" />
            </a>
            <h3>Green lizard</h3>
        </div>
        <div id="pic6">
            <img src="img/pic6.jpg" alt="Feeding the birds" />
            <a class="previous" href="#pic5">
                <img src="img/pic5.jpg" alt="Green lizard" />
            </a>
            <a class="next" href="#pic7">
                <img src="img/pic7.jpg" alt="A group of butterflies" />
            </a>
            <h3>Feeding the birds</h3>
        </div>
        <div id="pic7">
            <img src="img/pic7.jpg" alt="A group of butterflies" />
            <a class="previous" href="#pic6">
                <img src="img/pic6.jpg" alt="Feeding the birds" />
            </a>
            <a class="next" href="#pic8">
                <img src="img/pic8.jpg" alt="Ladybirds" />
            </a>
            <h3>A group of butterflies</h3>
        </div>
        <div id="pic8">
            <img src="img/pic8.jpg" alt="Ladybirds" />
            <a class="previous" href="#pic7">
                <img src="img/pic7.jpg" alt="A group of butterflies" />
            </a>
            <a class="next" href="#pic9">
                <img src="img/pic9.jpg" alt="Butterfly" />
            </a>
            <h3>Ladybirds</h3>
        </div>
        <div id="pic9">
            <img src="img/pic9.jpg" alt="Butterfly" />
            <a class="previous" href="#pic8">
                <img src="img/pic8.jpg" alt="Ladybirds" />
            </a>
            <a class="next" href="#pic10">
                <img src="img/pic10.jpg" alt="Trees in the mist" />
            </a>
            <h3>Butterfly</h3>
        </div>
        <div id="pic10">
            <img src="img/pic10.jpg" alt="Trees in the mist" />
            <a class="previous" href="#pic9">
                <img src="img/pic9.jpg" alt="Butterfly" />
            </a>
            <a class="next" href="#pic1">
                <img src="img/pic1.jpg" alt="Winter branches" />
```

```
                    </a>
                    <h3>Trees in the mist</h3>
                </div>
      </div>
    </div>
    </body>
    </html>
```

第7章 DIV+CSS布局

目前主流的布局方式就是DIV+CSS布局，这种布局方式与以前的table布局相比有很多的好处，比如有利于搜索引擎优化、代码精简、提高加载速度、样式便于操作、网页便于维护等等。本章就开始介绍如何使用DIV+CSS来布局我们的网页。



7.1 理解CSS与DIV定位

在学习DIV+CSS布局之前，我们先来看一些基本概念，例如对盒子模型的详细介绍和元素的定位，然后看一下如何给图片签名。

7.1.1 div与span标记

在HTML页面中，div和span标记都可以看做是一个盒子，盒子中内容可以是任意元素。如果标记中没有内容，那么这两个标记可以看做是空标记，在HTML页面中插入空标记不会对页面产生任何效果。

通常情况下，div元素被称为块元素，而span元素被称为内嵌元素。这是因为在没有设置样式的情况下，div元素需要单独占用一行，即使div元素中只有一个字符，也单独占用一行，span元素则根据内容的多少决定占用多宽的距离。例如下面这段代码，可以非常直观地展示div元素和span元素在这方面的区别，效果如图7.1所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>7.1</title>
<style type="text/css">
div{
border:solid 1px #470A0B;
margin-bottom:10px;
}
span{
border:solid 1px #2927D7;
```

```
}
</style>
</head>
<body>
<div>div中的内容</div>
<span>span中的内容</span>
</body>
</html>
```

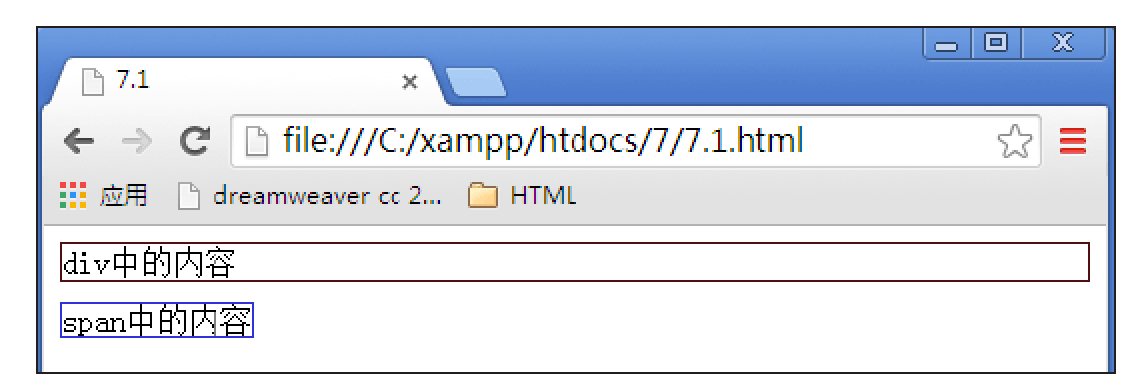


图7.1

在HTML中还有很多块级元素，它们都有一个共同的特点，那就是单独占用一行。它们的这个特点由display属性进行设置，如果display属性设置为block，则该元素就是一个块级元素；如果设置成inline，则该元素就是一个内嵌元素。所以，div元素和span元素还可以通过CSS样式设置转换器特点，例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>7.2</title>
<style type="text/css">
div{
border:solid 1px #470A0B;
display:inline;
}
span{
border:solid 1px #2927D7;
display:block;
margin-top:10px;
}
</style>
</head>
<body>
<div>div中的内容</div>
<span>span中的内容</span>
</body>
</html>
```

在这段代码中，我们将div元素的display属性设置为inline，将span元素的display属性设置为block。运行这段代码后，我们会发现div元素不再单独占用一行，而是根据div元素中内容的宽度进行缩放，而span元素则单独占用了一行，效果如图7.2所示。

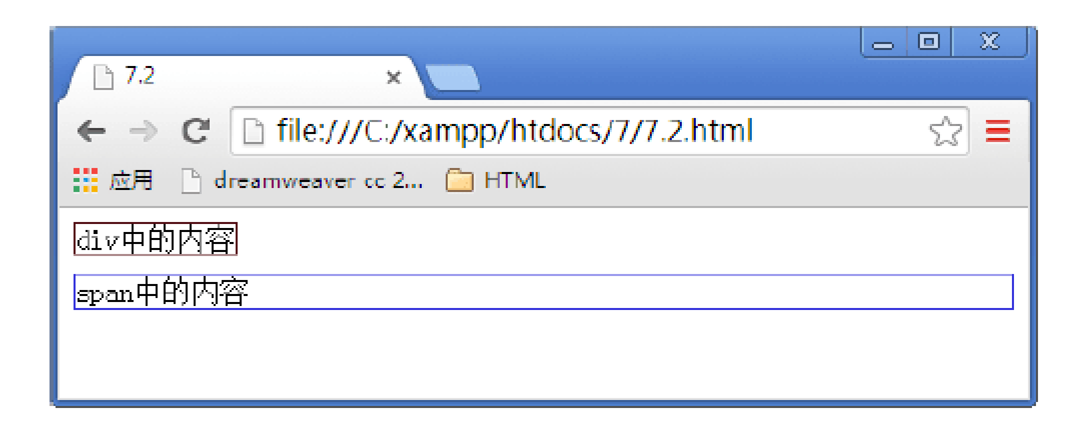


图7.2

7.1.2 盒子模型

盒子模型是CSS中一个重要的概念，理解了盒子模型才能更好地进行布局。盒子模型指定了元素如何显示以及如何相互交互，页面中所有元素都可以看成一个盒子。一个页面就是由很多个这样的盒子组成，这些盒子之间会相互影响，共同构成复杂的页面效果。

在CSS中，一个独立的盒子模型由margin（外边距）、border（边框）、padding（内边距）、content（内容）4个部分组成，如图7.3所示。

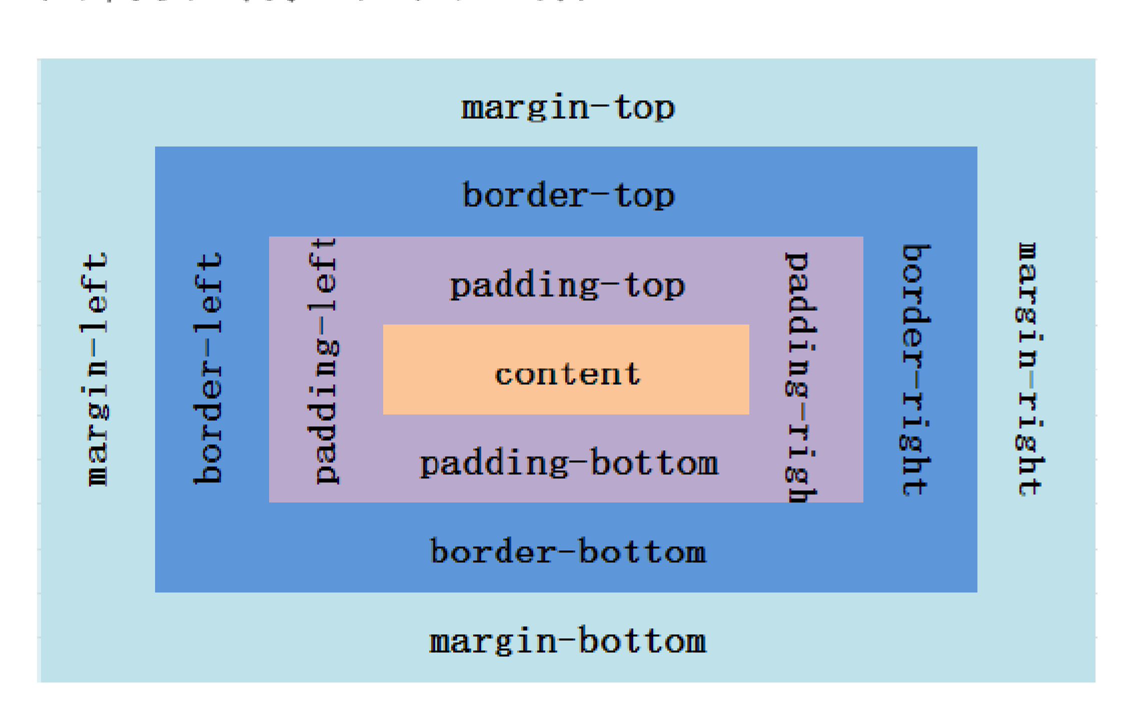


图7.3

1. 外边距

外边距是指容器边框以外的区域，经常用于设置两个元素之间的距离。在CSS中可以使用margin-top、margin-bottom、margin- left和margin-right属性分别设置外边距上下左右的距离，还可以使用margin属性按照上、右、下、左的顺序依次设置外边距。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>7.3</title>
<style type="text/css">
*{
margin:0;
}
div{
color:white;
```

```
background-color:#843637;
}
</style>
</head>
<body>
<div>外边距</div>
</body>
</html>
```

在这段代码中，只有一个div元素，首先使用通用选择器清除页面外边距，然后设置div元素中文本的样式和背景色，运行这段代码后，效果如图7.4所示。

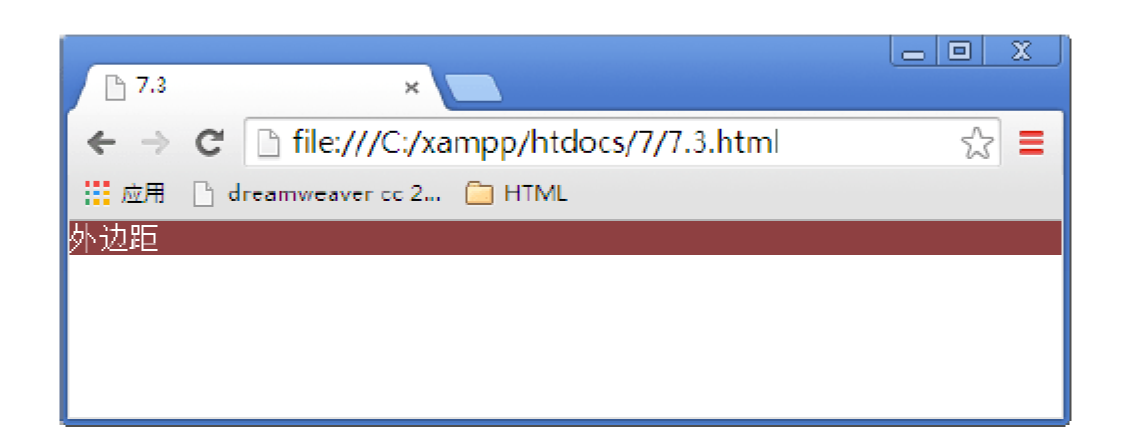


图7.4

可以看到，在清除div元素的外边距后，div元素与浏览器内容窗口的左边框、上边框和右边框紧紧相连。此时可以在div样式中添加如下代码，分别设置div元素的左外边距为50px、上外边距为100px、右外边距为50px，刷新页面后，效果如图7.5所示。

```
margin:100px 50px 0px 50px;
```

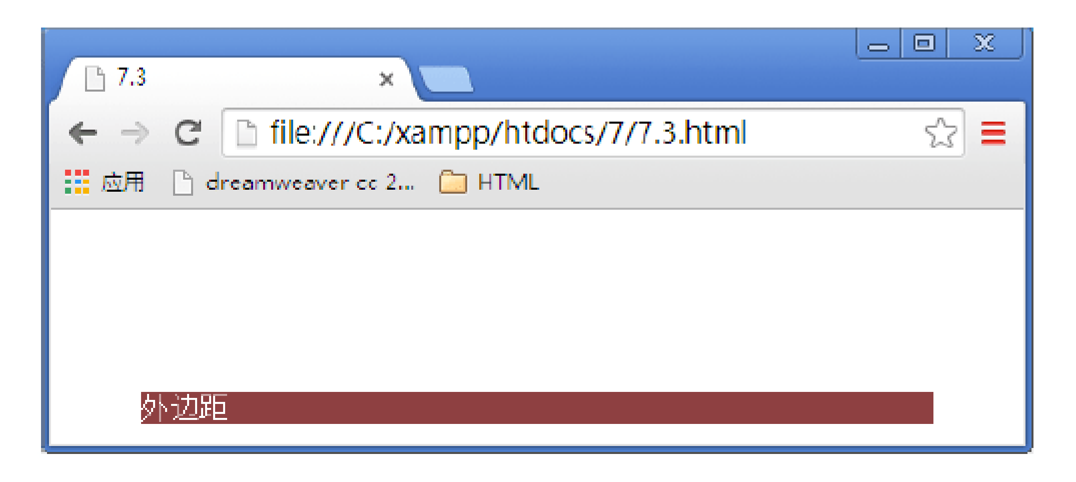


图7.5

还有一种情况，当上下外边距一样，左右外边距也一样时，还可以使用缩放的方式来设置外边距属性。例如，上下外边距都是10px，左右外边距都是15px时，可以使用以下代码进行设置。

```
margin:10px 15px;
```

2. 边框

边框是内容与填充的边界，只有设置了边框样式才能正确显示。在第6章中介绍表格和边框时我们已经介绍了边框的样式border-style，这些边框样式同样适用于盒子模型。

在CSS中可以使用border-width属性统一设置4条边框的宽度，还可以使用以下4个属性分别对这4条边框进行设置。

- border-top-width：设置上边框的宽度。

- border-right-width：设置右边框的宽度。
- border-bottom-width：设置下边框的宽度。
- border-left-width：设置左边框的宽度。

边框宽度的取值有3个，thin表示细的边框，medium表示默认的中等边框，thick表示粗的边框，效果如图7.6所示。

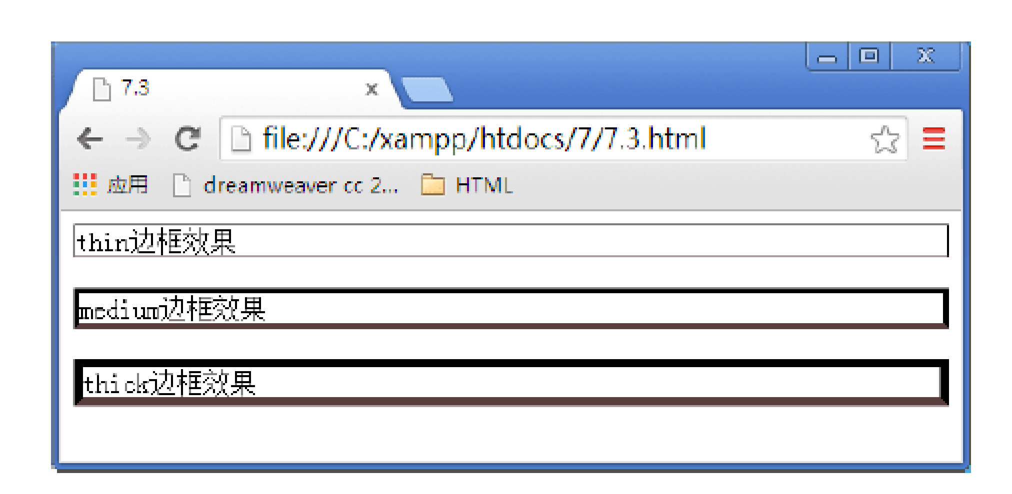


图7.6

边框的另外一个属性是border-color，用于设置边框的颜色。可以统一为4条边框设置颜色，也可以使用以下4个属性分别为它们设置颜色。

- border-top-color：设置上边框的颜色。
- border-right-color：设置右边框的颜色。
- border-bottom-color：设置下边框的颜色。
- border-left-color：设置左边框的颜色。

例如下面这段代码，分别为不同的边框设置不同的颜色，效果如图7.7所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>7.3</title>
<style type="text/css">
*{
margin:0;
}
div{
border:solid 10px #2819E8;
margin:10px;
border-top-color:red;
border-right-color:blue;
border-bottom-color:green;
border-left-color:yellow;
}
</style>
</head>
<body>
<div>边框颜色效果</div>
</body>
</html>
```

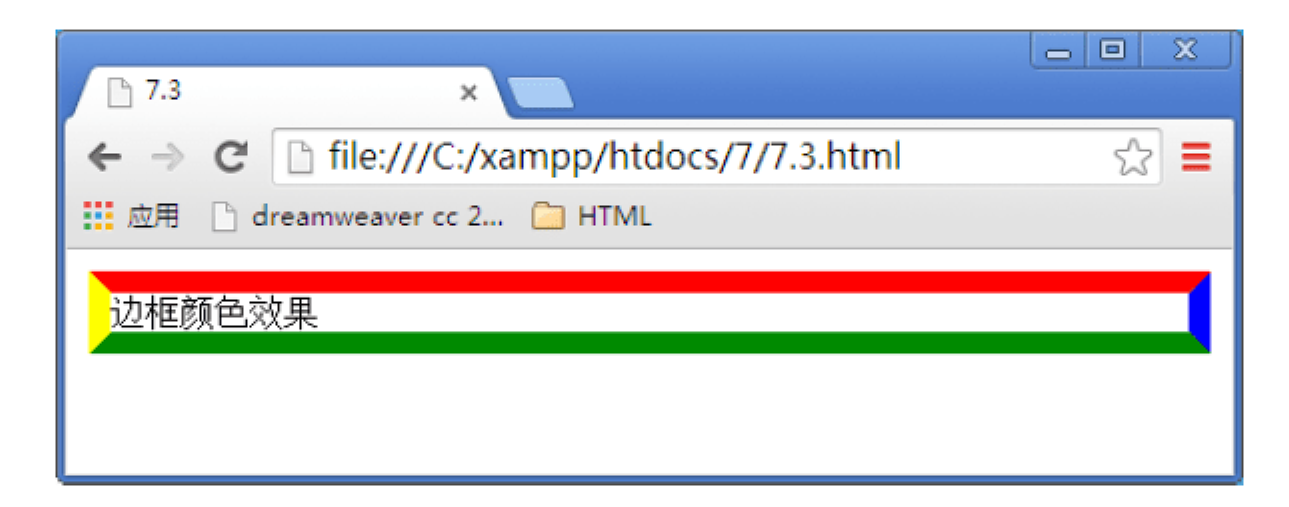


图7.7

还可以使用border-radius属性设置圆角边框，例如下面这段代码，设置边框的左上圆角和右上圆角半径为20px，左下圆角和右下圆角半径为5px，效果如图7.8所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>7.3</title>
<style type="text/css">
*{
margin:0;
}
div{
border:solid 2px #2819E8;
margin:10px;
font-size:36px;
border-radius:20px 20px 5px 5px;
}
</style>
</head>
<body>
<div>圆角边框效果</div>
</body>
</html>
```



图7.8

CSS中还提供了以下4种属性分别用于设置4条边框的圆角半径。

- border-top-left-radius：设置左上圆角半径。
- border-top-right-radius：设置右上圆角半径。
- border-bottom-left-radius：设置左下圆角半径。
- border-bottom-right-radius：设置右下圆角半径。

3. 内边距

内边距用于设置填充内容与边框之间的区域。与外边距相同，4条内边距可以使用padding统一设置，也可以分别使用padding-top、padding-bottom、padding-left和padding-right设置内边距。例如下面这段代码，第1个div元素没有设置内边距，而第2个div元素设置了内边距，效果如图7.9所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>7.3</title>
<style type="text/css">
*{
margin:0;
}
div{
border:solid 2px #2819E8;
margin:10px;
}
#paddingStyle{
padding-top:5px;
padding-bottom:10px;
padding-left:20px;
padding-right:30px;
}
</style>
</head>
<body>
<div>无内边距效果</div>
<div id="paddingStyle">有内边距效果</div>
</body>
</html>
```

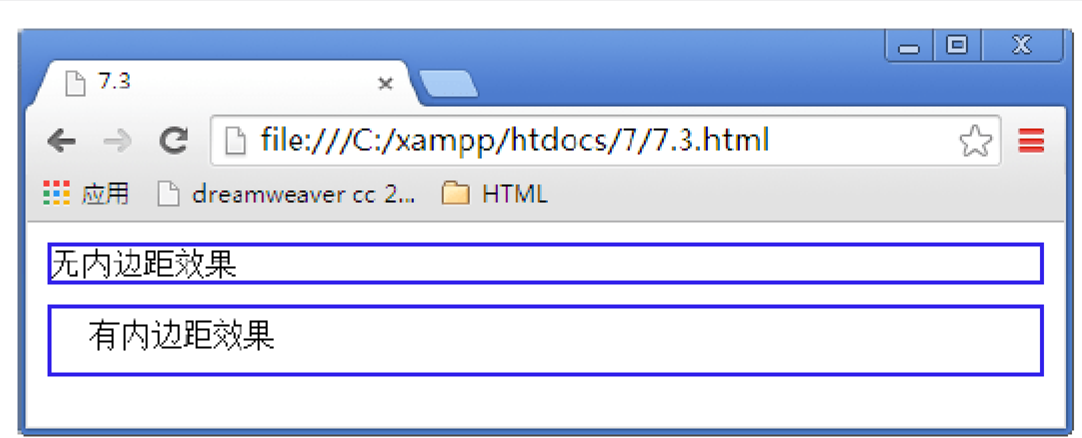


图7.9

4. 内容

内容是盒子模型中实实在在需要展示的东西，位于盒子模型的中心，可以根据不同的内容使用不同的CSS样式进行设置。如果内容是文字，就可以使用文本和段落相关的样式进行设置；如果内容是图像，就可以使用图像相关的样式进行设置。

7.1.3 元素的定位

在CSS中使用position属性可以设置元素在页面中的位置，即元素的定位。position属性有4种取值，分别介绍如下。

（1）static：默认定位方式，也称为文档流布局。
（2）relative：相对定位。以元素原来所在位置为基点，重新定位元素的位置。
（3）absolute：绝对定位。彻底清除元素原来所在位置，重新为元素设置新的位置，并且不依赖任何对象。
（4）fixed：将元素相对于窗口固定位置。

我们先来看一个相对定位的例子。页面中有3个div元素，第1个div元素不设置定位的样式；第2个div元素设置相对定位，并设置left属性为-50px；第3个div元素也设置相对定位，left属性设置为50px。代码如下：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>7.4</title>
<style type="text/css">
div{
border:solid 2px #2819E8;
margin:10px;
background-color:#721E20;
color:white;
}
#posA{
position:relative;
left:-50px;
}
#posB{
position:relative;
left:50px;
}
</style>
</head>
<body>
<div>原来的位置</div>
<div id="posA">新的位置A</div>
<div id="posB">新的位置B</div>
</body>
</html>
```

运行这段代码后，可以看到第2个div元素已经向左移动到浏览器窗口的外边，而第3个div元素向右移动到浏览器窗口的外边。同样是设置left属性，出现两种不同效果的原因是一个取值为负数，一个取值为正数，效果如图7.10所示。

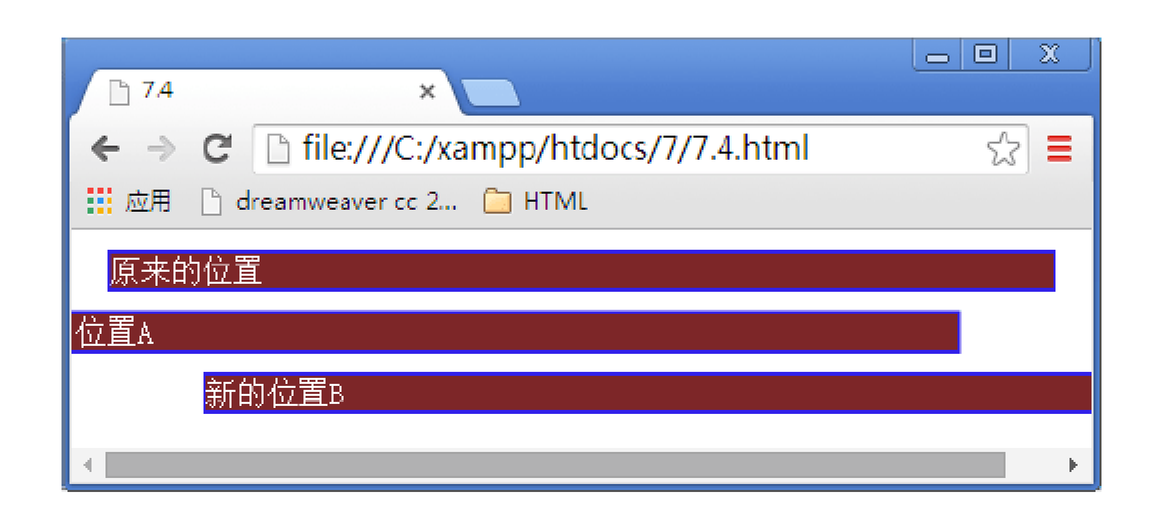


图7.10

修改第2个div和第3个div的样式，让两个div以绝对定位的方式重新设置位置，代码如下所示：

```
#posA{
position:absolute;
left:50px;
top:70px;
}
#posB{
position:absolute;
left:200px;
top:70px;
}
```

刷新页面后的效果如图7.11所示。

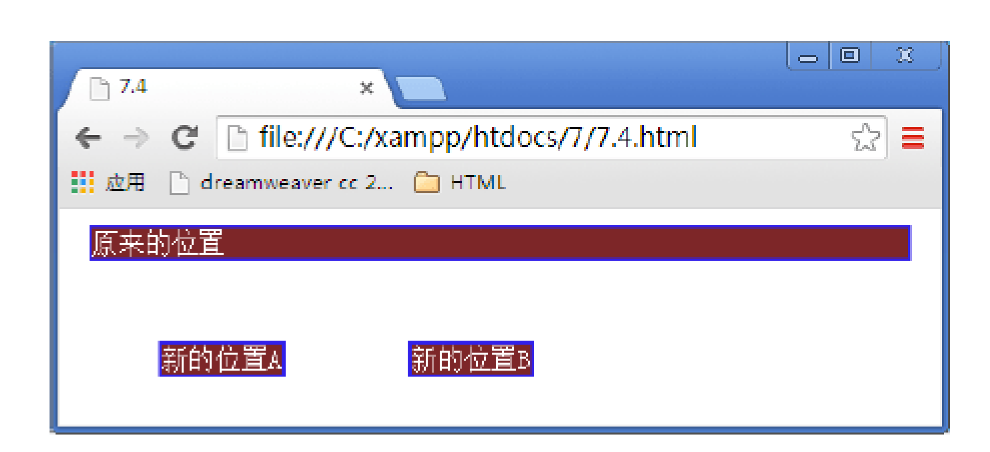


图7.11

重新设置第2个div的高度，使浏览器出现上下滚动条，然后设置第3个div的position为fixed，相关代码如下：

```
#posA{
position:absolute;
left:50px;
top:70px;
height:400px;
}
#posB{
position:fixed;
left:200px;
top:70px;
}
```

运行这段代码后，当滚动鼠标时，第3个div的位置始终保持固定不变，而前两个div的位

置会随着鼠标的滚动而移动，如图7.12所示。

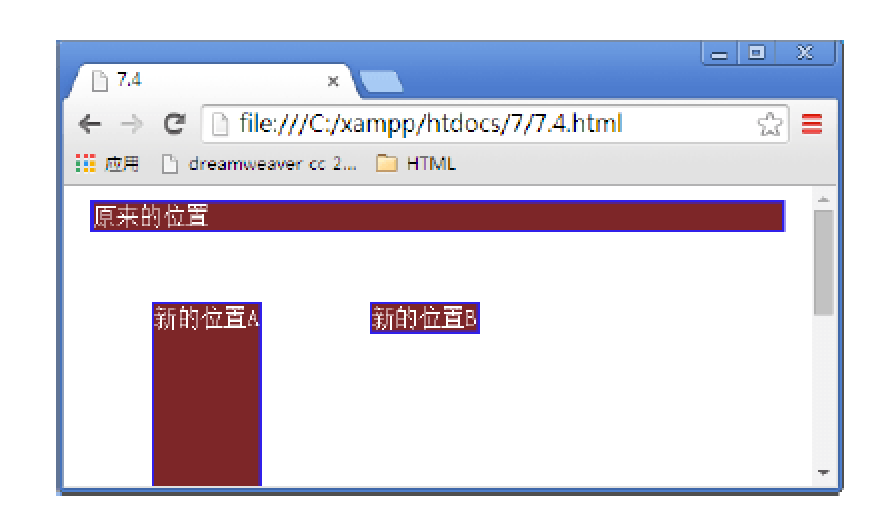


图7.12

7.1.4 给图片签名

如果你的想象力够丰富，就可以使用CSS创建各种有趣的页面效果，例如给图片签名。下面我们来分析一下实现这个效果的关键步骤。

首先可以确定一点，文本必须和图片显示在同一个位置上。这就好比两个图层叠加在一起，下层是图片，上层是文字签名。要实现这种效果，可以将图片和文本分别放在两个div元素中，然后通过设置div元素的float属性使文本浮动到图片上。

其次，要控制文字在图片上显示的位置。要对文字进行定位处理，这就需要用到CSS的position属性。

最后，我们要让签名显示的更好看一些，就需要设置文本的颜色、大小、样式和字体等。

通过以上的分析，有了以下代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>7.5</title>
<style type="text/css">
#img{
float:left;
}
#txt{
color:red;
position:absolute;
left:220px;
top:210px;
font-size:64px;
font-weight:bold;
font-family:"华文行楷";
font-style:oblique;
}
</style>
</head>
<body>
<div id="img"><img src="img/img02.jpg" /></div>
```

```
<div id="txt">汪</div>
</body>
</html>
```

运行这段代码后，效果如图7.13所示。

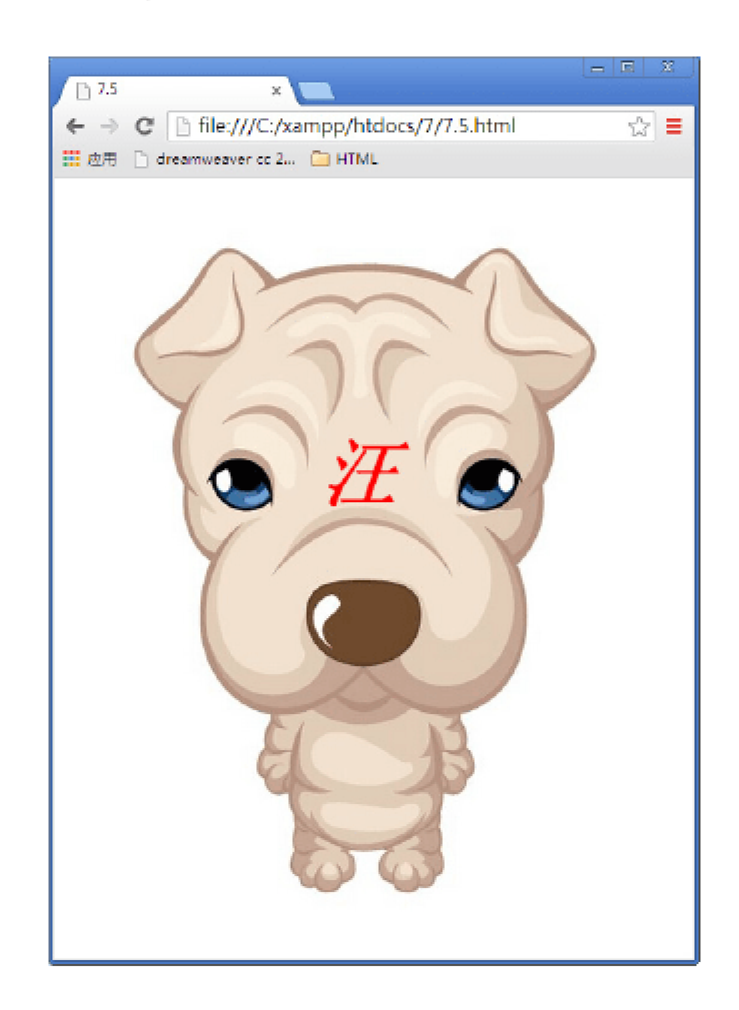


图7.13



7.2 DIV+CSS网页布局方法

DIV+CSS是目前主流的网页布局方法。可以将div看成是一个容器，容器中可以放文字、图像、表格等其他元素，通过CSS的控制就能实现各种网页效果。

7.2.1 div的并列与嵌套结构

在分析网页结构的时候，我们可以将网页中的各个区域划分为一个一个的div容器。比如一般的网页都会有头、内容和底部，我们可以用3个div元素分别表示这3个区域，代码如下所示：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>7.6</title>
<style type="text/css">
*{
margin:0;
padding:0;
```

```
}
div{
border:solid 1px #B06317;
text-align:center;
font-weight:bold;
}
#header{
height:10%;
min-height:100px;
background-color:#C9E5C4;
}
#content{
height:80%;
min-height:400px;
background-color:#D5C582;
}
#foot{
height:10%;
min-height:100px;
background-color:#8FA3E3;
}
</style>
</head>
<body>
<div id="header">网页的头部</div>
<div id="content">网页的内容</div>
<div id="foot">网页的底部</div>
</body>
</html>
```

在这段代码中，依次排列了3个div元素。我们设置第1个和第3个div元素的高度为10%，最小高度为100px，第2个div元素的高度为80%，最小高度为400px。运行这段代码后，效果如图7.14所示。

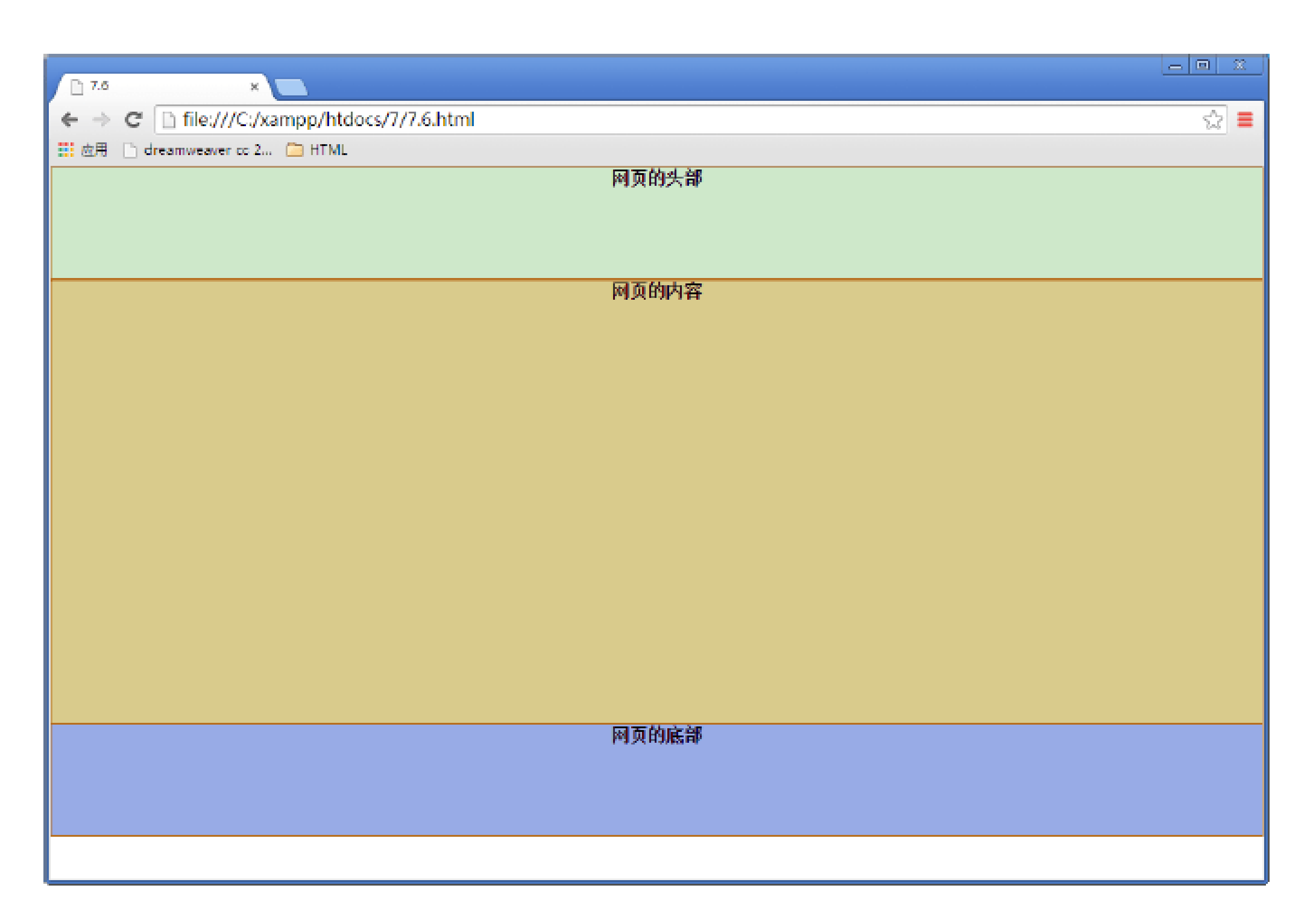


图7.14

有时候我们不仅要纵向排列div元素，还需要横向排列div元素。由于div元素是块级元素，每个div元素都单独占据一行，为了达到横向排列div元素的目的，可以通过设置float属性来控制div的排列方向。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>7.7</title>
<style type="text/css">
*{
margin:0;
padding:0;
}
div{
border:solid 1px #B06317;
text-align:center;
font-weight:bold;
float:left;
min-height:400px;
}
#left{
width:10%;
min-width:100px;
background-color:#C9E5C4;
}
#center{
width:70%;
min-width:300px;
background-color:#D5C582;
}
#right{
width:10%;
min-width:100px;
background-color:#8FA3E3;
}
</style>
</head>
<body>
<div id="left">左边的区域</div>
<div id="center">中间的区域</div>
<div id="right">右边的区域</div>
</body>
</html>
```

在这段代码中，仍然依次排列了3个div元素。通过CSS样式设置第1个和第3个div元素的宽度为10%，最小宽度为100px，第2个div元素的宽度为70%，最小宽度为300px。然后在div选择器中设置div元素的float属性为left，让3个div元素向左浮动。运行这段代码后，效果如图7.15所示。

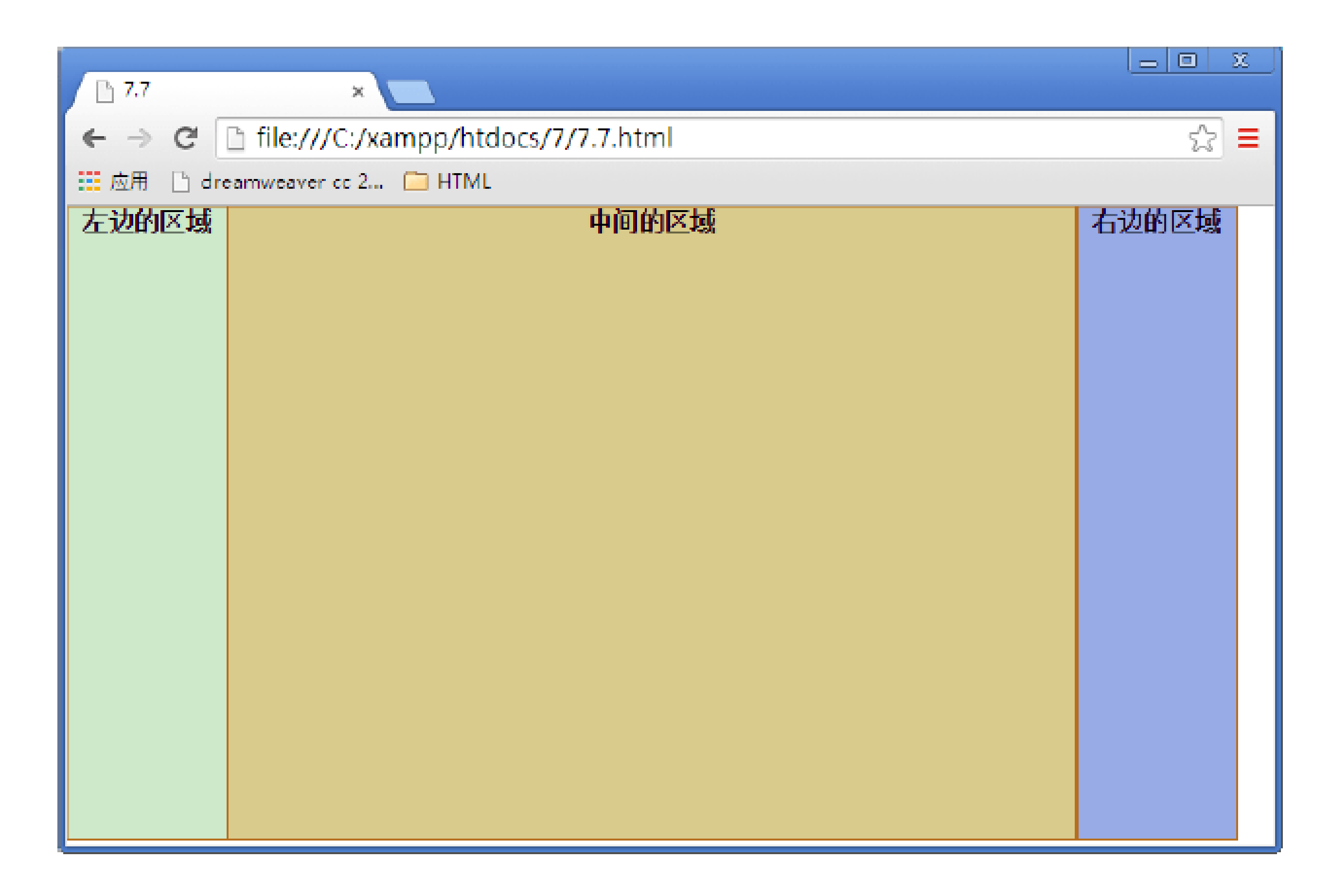


图7.15

为了实现某些特殊的效果，我们经常会使用嵌套的div来布局HTML页面，并通过CSS设置完成这些特殊的效果。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>7.8</title>
<style type="text/css">
*{
margin:0;
padding:0;
}
#Container{
height:500px;
text-align:center;
font-size:16px;
font-weight:bold;
}
#left{
float:left;
height:500px;
width:150px;
margin-right:5px;
background-color:#DCD595;
}
#right{
margin-left:155px;
}
#top{
background-color:#A6C6D8;
margin-bottom:5px;
height:100px;
}
#bottom{
```

```
height:395px;
}
#contentLeft{
width:600px;
float:left;
height:395px;
background-color:#93E77F;
}
#contentRight{
margin-left:605px;
height:395px;
background-color:#C288DC;
}
</style>
</head>
<body>
<div id="Container">
   <div id="left">左侧栏</div>
   <div id="right">
        <div id="top">顶部栏</div>
        <div id="bottom">
            <div id="contentLeft">内容左</div>
            <div id="contentRight">内容右</div>
        </div>
   </div>
</div>
</body>
</html>
```

在这段代码中，最外层的div元素作为最大的容器，内嵌了两个div元素，这两个div元素将设置为左右排列；右侧的div元素中又内嵌了两个div元素，这两个div元素将设置为上下排列；下面的这个div元素中又内嵌了两个div元素，这两个div元素又会设置成左右排列。这些div元素经过一层一层的嵌套，最终通过CSS样式设置为如图7.16所示效果。



图7.16

7.2.2 固定高度布局

在使用div+css布局页面的时候，为div元素设置固定的高度，是布局过程中经常会使用的一种方法。使用固定高度布局页面时，需要准确计算每个div的高度。如果为div元素设置了边框、内边距或外边距，此时仍需要将这些宽度计算在相应的高度中，否则整个页面中的div元素将会在水平方向上相差几个像素。如图7.16所示，如果给#left选择器和#contentLeft选择器添加以下代码：

```
border:solid 1px #DC2E74;
```

在不修改高度的情况下，页面底部就会出现几个像素的差距，效果如图7.17所示。这是因为设置div边框宽度为1px后，在垂直方向上顶部栏和内容左分别新增加了2px的边框高度，所以#right选择器的高度与#left选择器的高度就相差了4px，而内容左与内容右的高度也相差了2px。

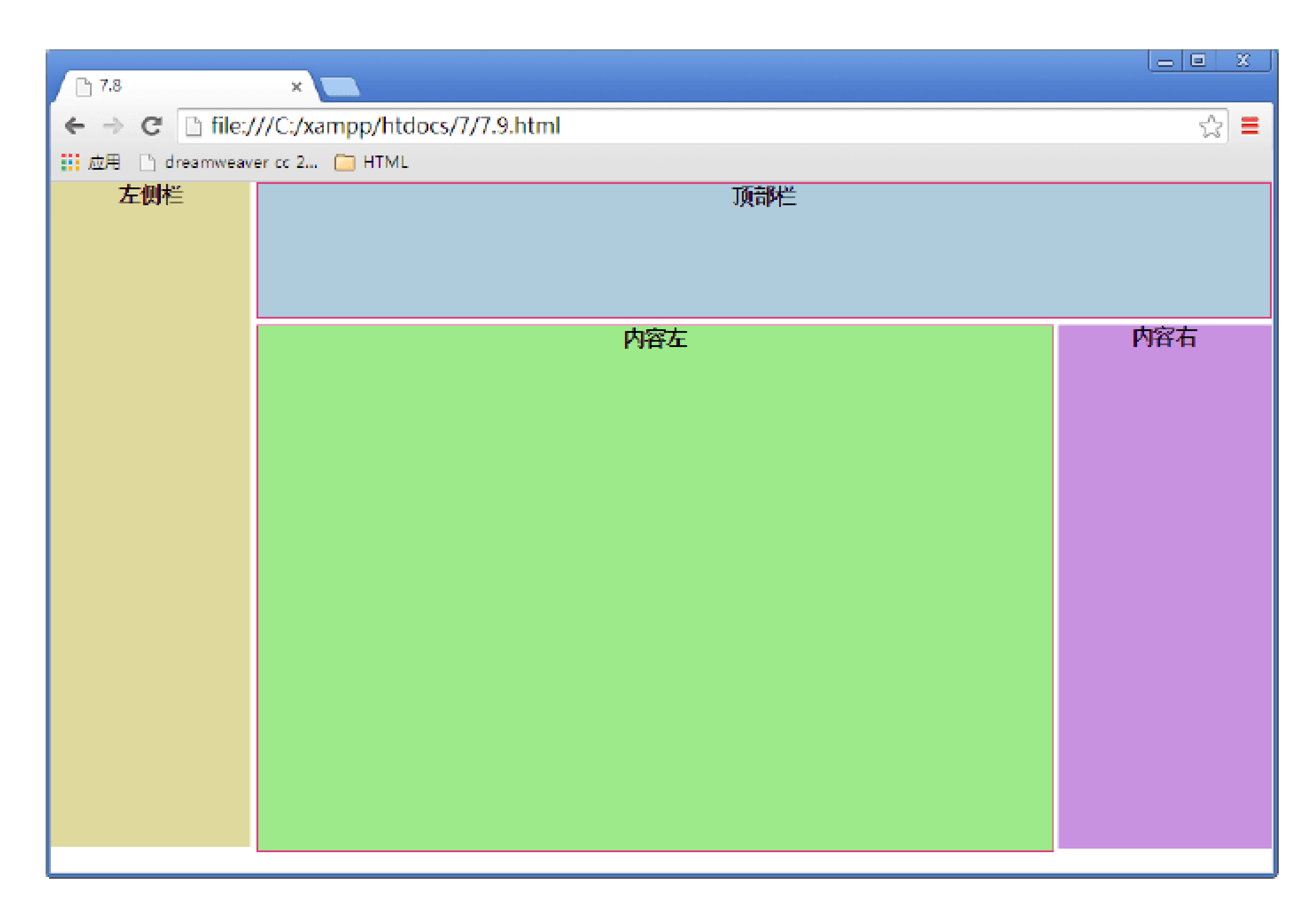


图7.17

7.2.3 自适应高度布局

固定高度布局常用于页面中内容比较固定的区域，如页面顶部的logo区域以及页面底部的友情链接这些区域。在内容区域中，往往不建议使用固定高度布局，这是因为这些区域的内容根据实际情况可能有变化，如果内容的高度超出了div元素设置的高度，就无法实现我们期望的效果。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>7.10</title>
<style type="text/css">
#Container{
height:300px;
```

```
background-color:#C89899;
}
#dataRow{
height:100px;
background-color:#8BBDB2;
}
</style>
</head>
<body>
<div id="Container">
   <div id="dataRow">
        数据行<br>
        数据行<br>
        数据行<br>
        数据行<br>
        数据行<br>
        数据行<br>
        数据行<br>
        数据行<br>
        数据行<br>
    </div>
</div>
</body>
</html>
```

在这段代码中，设置外层div元素的高度为300px，内层div元素的高度为100px。分别为两个div元素设置不同的背景，当内层div元素中的内容行太多时，超出部分内容的背景色将会显示为外层div元素的背景色，这并不是我们希望看到的，如图7.18所示。



图7.18

为了避免这种情况的出现，我们需要使用自适应高度布局的方式进行修改。修改方法很简单，只需要将# dataRow选择器中的height属性用min-height属性替换即可。刷新页面后，我们可以看到一个自适应高度布局的效果，如图7.19所示。

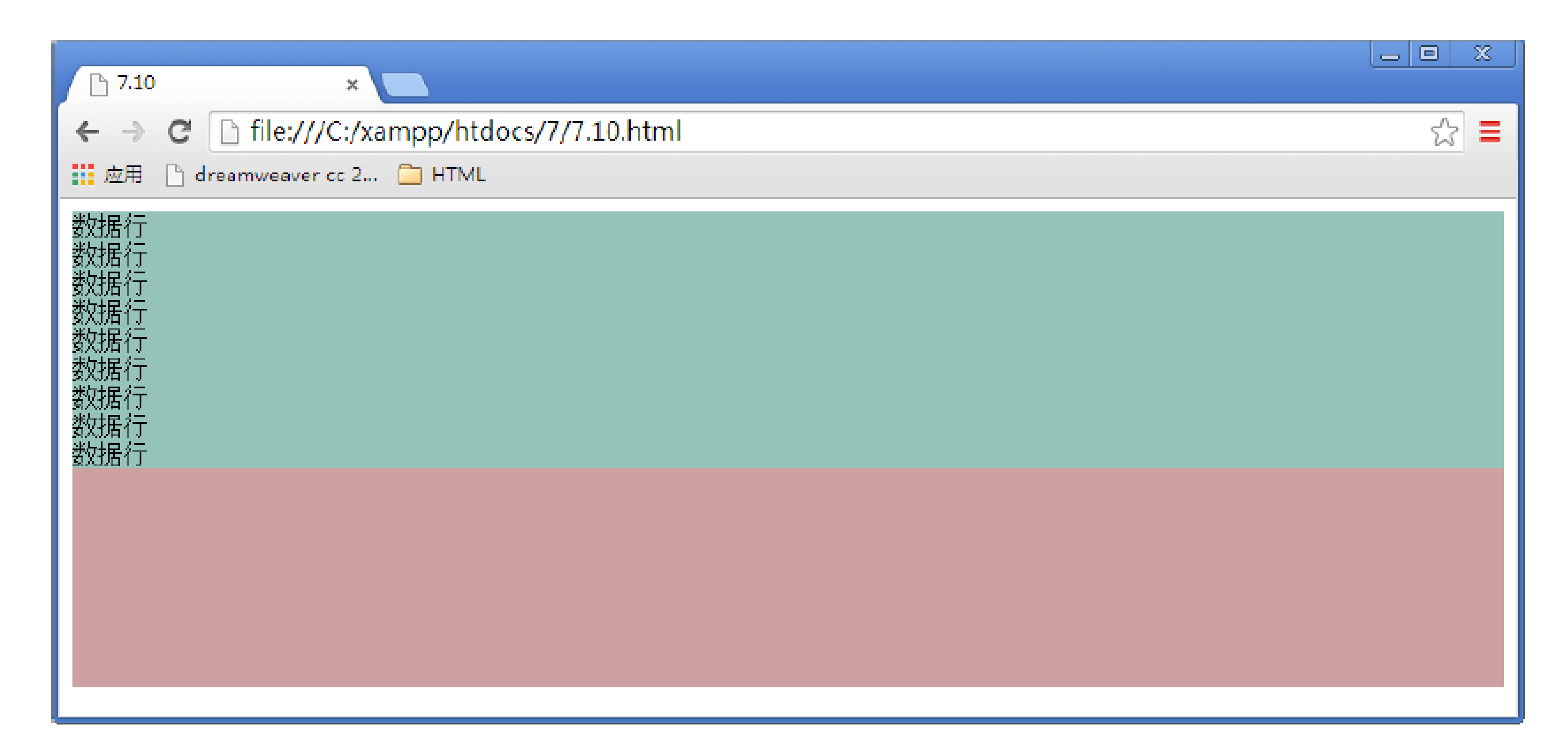


图7.19

7.2.4 多行多列布局

使用div+css还可以设置多行多列的页面布局，常见的案例有商品浏览页面、图像浏览页面等。在创建多行多列布局之前，首先要规划页面结构，确定要创建几行几列的页面布局，这样才能够确定设计使用多少个div元素。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>7.11</title>
<style type="text/css">
#Container{
height:310px;
float:left;
background-color:#C89899;
}
#data{
height:100px;
width:100px;
float:left;
margin-right:5px;
margin-bottom:5px;
background-color:#8BBDB2;
}
</style>
</head>
<body>
<div id="Container">
    <div id="data">数据</div>
    <div id="data">数据</div>
    <div id="data">数据</div>
    <div id="data">数据</div>
    <div id="data">数据</div>
```

```
        <div id="data">数据</div>
        <div id="data">数据</div>
        <div id="data">数据</div>
    </div>
    </body>
    </html>
```

在这段代码中，总共有8个div元素用于演示多行多列布局的效果。使用CSS设置外层容器向左浮动，然后设置内嵌div元素也向左浮动，为它们设置不同的背景色、高度和宽度，效果如图7.20所示。如果改变浏览器窗口的大小，这些div元素也会自动适应浏览器窗口大小，从而改变它们的排列方式，效果如图7.21所示。

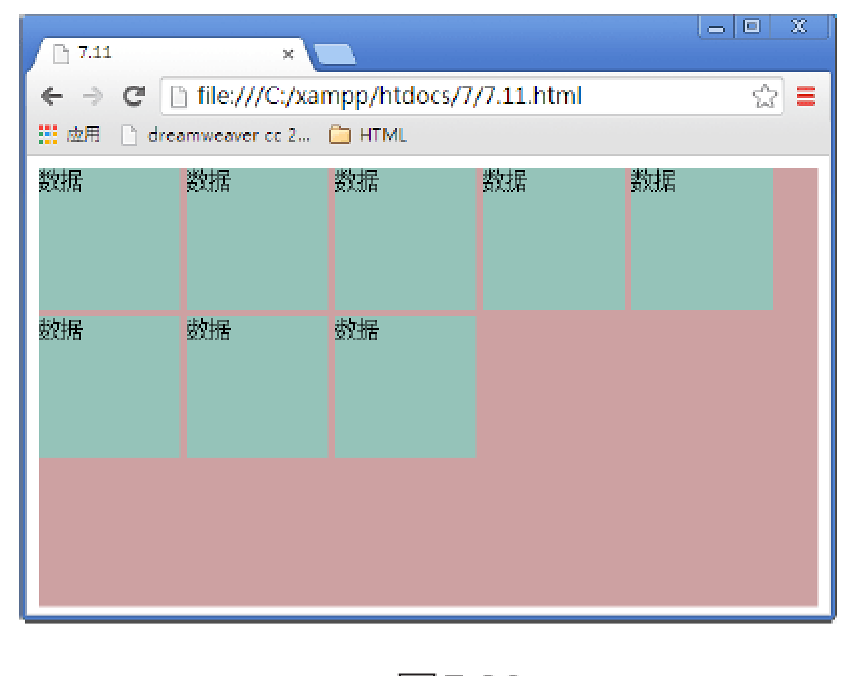


图7.20

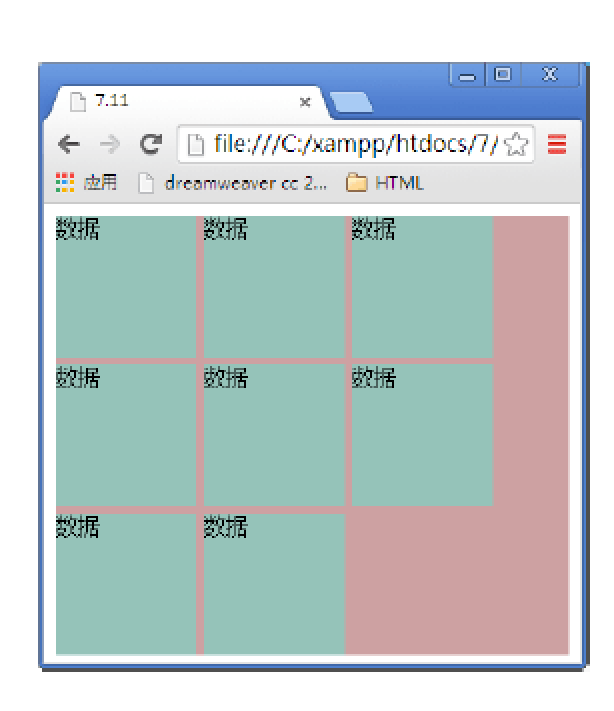


图7.21



7.3 页面布局设计

页面布局是网页设计中非常重要的一个环境。早期大量的页面布局都使用table布局，随着技术的发展，目前已经很少有人用table布局页面了，取而代之的是div+css布局。本节将制作一个固定宽度的页面布局效果，帮助大家掌握div+css的布局方法。详细制作步骤如下：

01 新建一个HTML文件，在代码视图中编写一个div元素，设置id属性为container，然后使用通配符*清除所有元素内外边距，使用id选择器设置这个div居中显示，宽度为900像素。相关代码如下所示：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>无标题文档</title>
<style>
* {
margin: 0;
padding: 0
```

```
}
#container {
margin: 0 auto;
width: 900px;
}
</style>
</head>
<body>
<div id="container">
</div>
</body>
</html>
```

因为没有给这个div设置任何高度和边框宽度，也没有在这个div内编写任何内容，所以这段代码在浏览器中没有任何效果。这里设置div的margin属性，仅仅是为了让这个div容器以及它里面的内容能够在水平方向上居中显示，这样当用户改变浏览器窗口大小的时候，能够保证页面居中显示。

02 在步骤**01**创建的div内再嵌套4个div，分别用于显示页面的头部、菜单、内容和底部，其中内容div中再嵌套两个div，分别用于显示侧边栏和内容区域。相关代码如下所示：

```
<body>
<div id="container">
    <div id="header">这里是页面头部，用于显示网站的logo和网站名称</div>
    <div id="menu">这里是菜单栏，用于显示页面导航</div>
    <div id="mainContent">
        <div id="sidebar">这里是侧边栏，用于显示页面导航或者其他信息</div>
        <div id="content">这里是页面内容区域</div>
    </div>
    <div id="footer">这里是页面底部，可用于显示页面导航、友情链接和版权等信息</div>
</div>
</body>
```

这段代码在浏览器中的显示效果如图7.22所示。



图7.22

03 设置页面顶部样式，高度为100像素，底部外边距为5个像素，并设置背景色，相关代码如下所示：

```
#header {
height: 100px;
background: #C8D9D7;
margin-bottom: 5px;
}
```

刷新页面后，效果如图7.23所示。



图7.23

04 设置菜单样式，菜单高度为30像素，底部外边距为5个像素，并设置背景颜色，相关代码如下：

```
#menu {
height: 30px;
background:#C6ABAC;
margin-bottom: 5px;
}
```

刷新页面后，效果如图7.24所示。

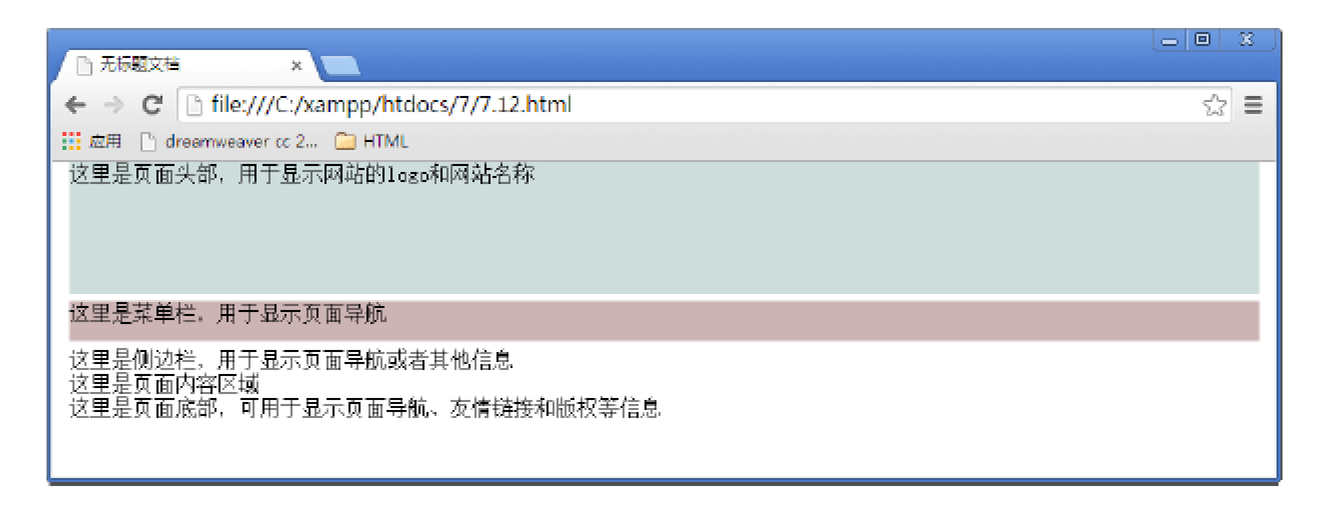


图7.24

05 接下来设置内容区域的样式，首先设置内容区域外边id属性为mainContent的样式，设置高度为500像素，底部外边距为5个像素；再设置侧边栏的样式，设置侧边栏向右浮动，宽度为200像素，高度为500像素，并设置背景色；最后设置内容区域的样式，设置内容区域向左浮动，宽度为695像素，高度为500像素，并设置背景色。相关代码如下所示：

```
#mainContent {
height: 500px;
margin-bottom: 5px;
}
#sidebar {
float: right;
width: 200px;
height: 500px;
background:#99E6B5;
}
#content {
float: left;
width: 695px;
```

```
height: 500px;
background:#E19EE3;
}
```

需要注意的是，在步骤01中我们设置了页面宽度为900像素，侧边栏的宽度为200像素，那么内容区域的宽度应该是700像素。由于页面的顶部和菜单都设置了底部外边距为5个像素，所以这里为了保持页面整体效果，让内容区域的宽度缩小了5个像素。刷新页面后的效果如图7.25所示。



图7.25

06 最后设置底部的样式，设置底部高度为60个像素，并设置背景色，相关代码如下所示：

```
#footer {
height: 60px;
background:#5F725E;
}
```

刷新页面后的最终效果如图7.26所示。



图7.26

07 通过本节的介绍，有兴趣的同学还可以尝试制作更多的页面布局效果。

第8章 制作网站导航菜单

导航菜单是网站中非常重要的内容，它是整个网站的地图。网站中主要的界面都应该设置导航菜单，导航菜单中不需要包含网站所有页面的目录，但必须包括网站主要内容页面目录。



8.1 网站导航菜单概述

网站导航菜单是网页设计中十分重要的部分，它为用户提供了浏览整个网页的快速通道。下面我们将简单介绍网站导航菜单的作用以及制作标准。

8.1.1 网站导航菜单的作用

网站导航的目的就是帮助用户找到他们需要的信息，可以概括为以下几点：

（1）引导用户完成网站各内容页面的跳转。这个功能是最常见的，全局导航、局部导航和辅助导航等都是为了引导用户浏览相关的页面。

（2）整理网站中各页面之间的关系。为网站中的内容建立一个索引，这个最常见的应用就是网站地图和内容索引表，展现了整个网站的目录信息，帮助用户快速找到相应的内容。

（3）定位用户在网站中的位置。这个在面包屑导航中得到了充分的体现，它帮助用户识别当前浏览的页面与网站整体内容关系，以及与网站中其他内容的联系和区分。

8.1.2 网站导航菜单的制作标准

网站导航菜单是网站中每个网页的重要内容，导航菜单中主要包含了网站的主页、主要功能、联系方式等一些用户感兴趣的内容，这些内容应该与网站结构图中的主要目标相关联。在制作网站导航时应注意以下几点：

（1）主页上的导航菜单一定要清晰，醒目。主页中的导航是用户浏览网站的第一个入口，一般设计为一级目录。通过主页中的导航用户和蜘蛛都可以深入访问到网站所有重要内容，主页上的导航栏目通常采用文本链接。

（2）网站导航菜单要用文本做链接。虽然图片和文本都可以用来实现导航菜单，但是用文本会更好些。很多网站上的导航菜单都使用图片或flash按钮做链接，这样虽然可以让网站看起来更漂亮和美观，但是并不利于网站的推广，搜索引擎很难搜索到这类网站，所以网站导航一定要用文本做链接。

（3）创建面包屑导航。面包屑导航可以让用户了解当前所处位置，以及当前页面在这个网站中的位置，体现了网站的架构层级，能够帮助用户快速学习和了解网站内容和组织方式，从而形成很好的位置感。面包屑导航还可以提供返回各个层级的快速入口，方便用户操作。

（4）重要内容应该出现在首页。除了主栏目，还应该将次级目录中的重要内容以链接的方式在首页或其他子页中多次呈现，以突出这些内容为重点。搜索引擎会对这种一站内多次出现的链接给予充分重视，对网页级别提高有很大帮助，这也是一般网站首页的网页级别高于其他页面级别的重要因素，因为每个子页都对首页进行了链接。



网站导航菜单的种类

随着互联网技术的不断发展以及设计者个性化的充分展现，网站导航的设计已经决定了网站的风格。根据目前互联网上出现的各类导航菜单，可以将其分为以下6种类型。

1. 三维导航设计

平面的导航设计越来越显得平庸和乏味，更多的设计者喜欢用三维立体效果来展现导航设计，让导航看起来更有立体感。图8.1所示为拥有三维立体效果的网页。

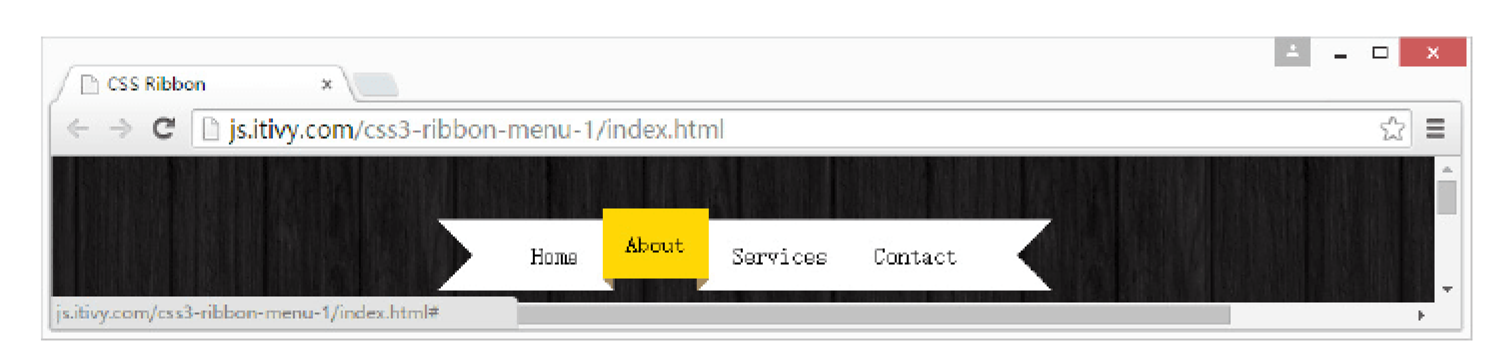


图8.1

2. 说话气泡导航设计

把菜单设计成讲话的气泡形式，似乎是另一种流行的趋势。图8.2为拥有说话气泡导航的网页。

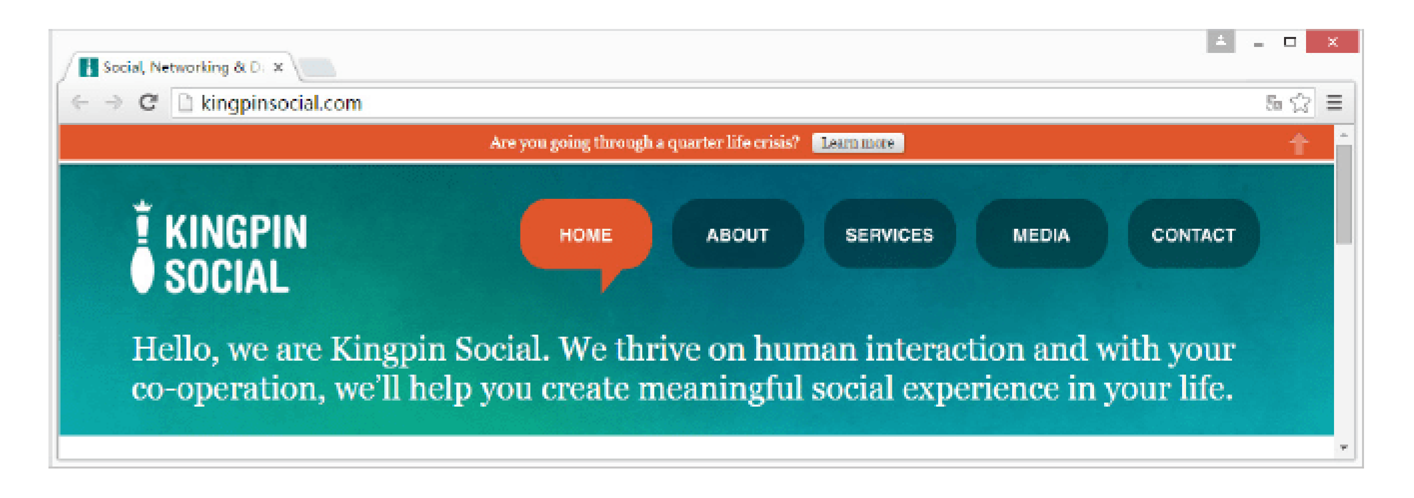


图8.2

3.圆角导航设计。

圆角经常用来软化规整的矩形，按钮的外观是为了吸引用户单击他们。图8.3为拥有圆角导航的网页。

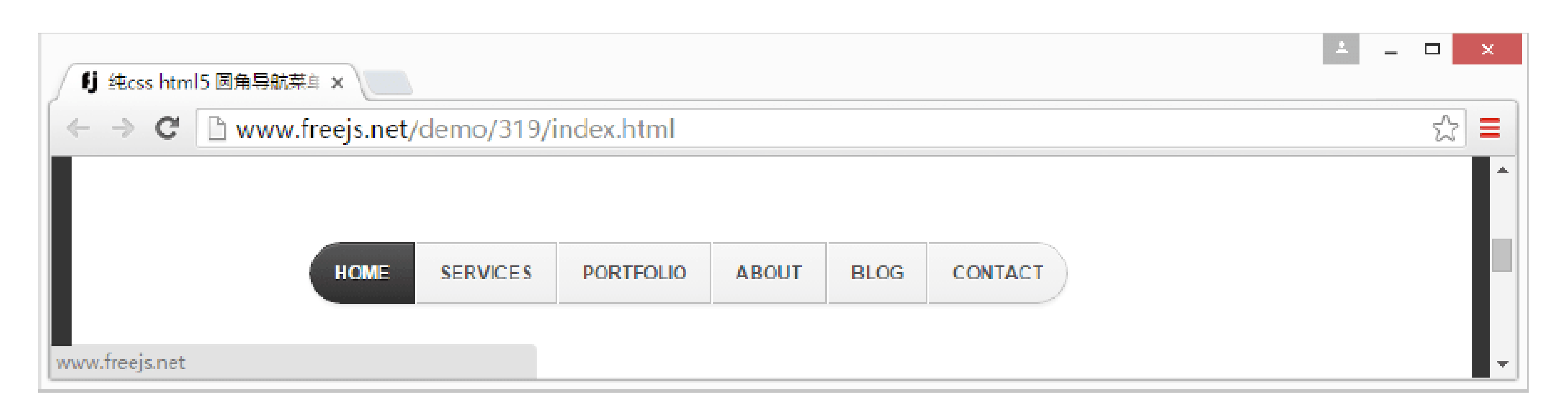


图8.3

4. 应用图标的导航设计

因为现在带宽不再令人担心了，所以越来越多的精致图标被应用到导航设计中。由于视觉上的吸引力，使用图标的人越来越多，这种趋势仍在继续。图标不仅能吸引眼球，还有助于用户进行视觉识别。图8.4为拥有图标导航的网页。

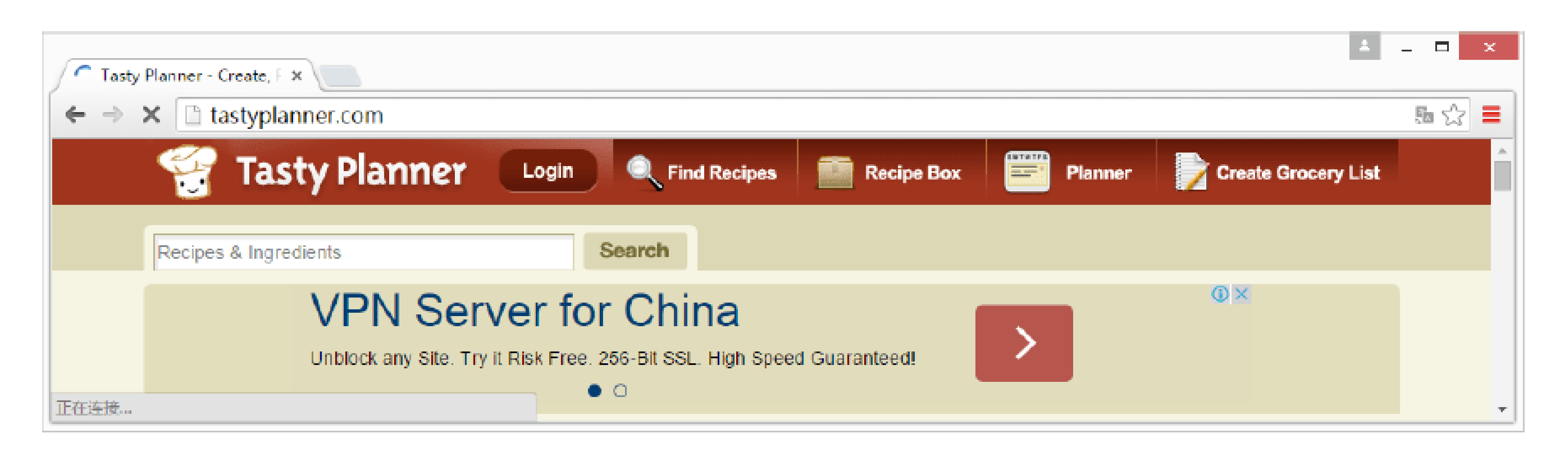


图8.4

5. JavaScript动画

JavaScript技术使Web设计人员只用几行代码的网页元素就可以创建动画，设计师们最近一直在使用功能多用又美观的JavaScript。图8.5为导航上添加了动画的网页。

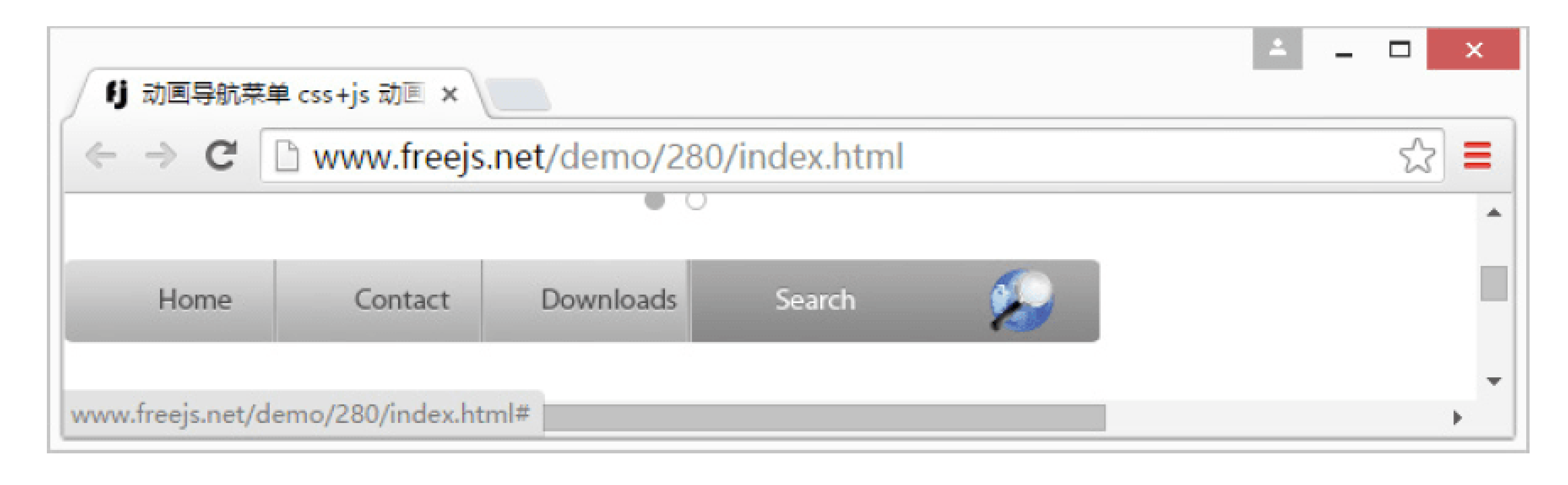


图8.5

6. 不规则形状导航设计

由于大多数网站都用的是直边和尖角，不规则的形状让你有机会摆脱俗套。图8.6为使用了不规则形状导航的网页。

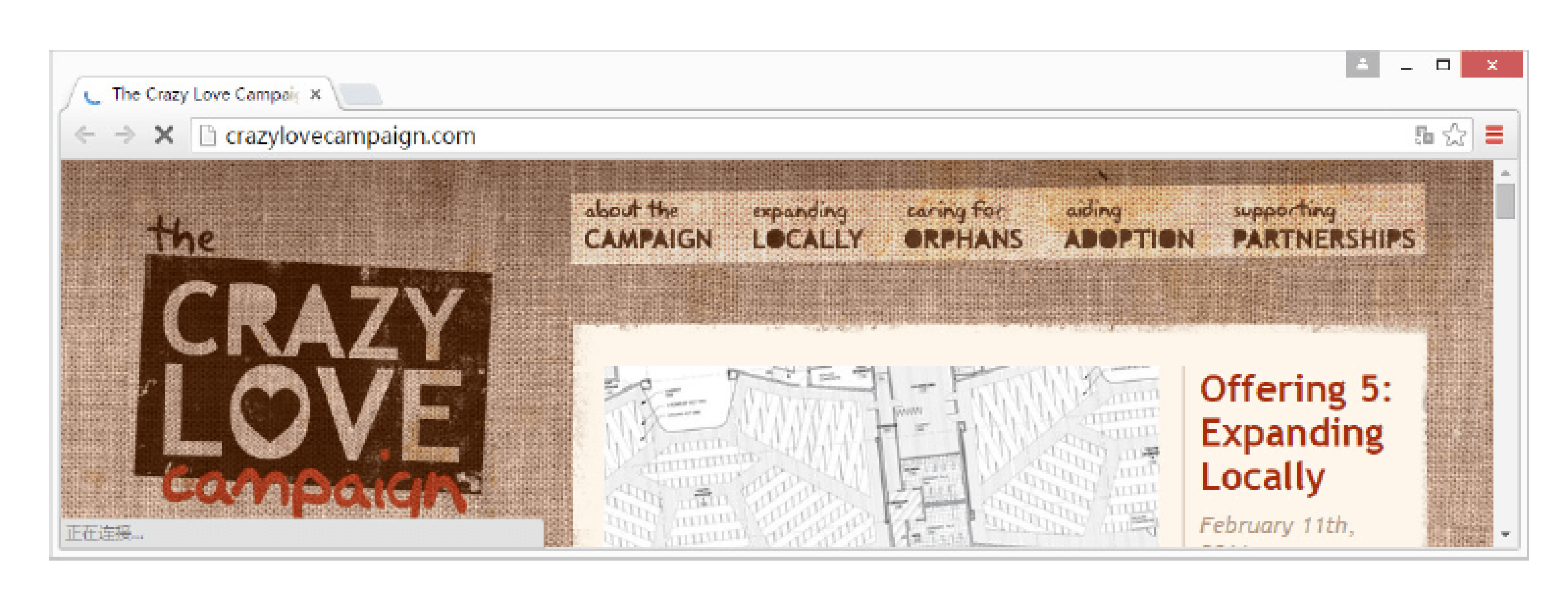


图8.6



8.3 创建翻转按钮

翻转按钮是一种用来制作网站导航菜单的素材。当用户把鼠标停放在翻转按钮上时，按钮会呈现一个翻转的动作，同时变换成另一种样式；当用户把鼠标从翻转按钮上移开时，按钮又会恢复到原来的样式。本节将介绍几种创建翻转按钮的方法。

8.3.1 用代码创建翻转按钮

CSS 中的transition属性用于设置CSS在一定的时间区间内平滑过渡的属性值，这种效果可以在鼠标单击、获得焦点、被点击或元素有任何改变时触发，并圆滑地以动画效果改变CSS的属性值。它的语法如下所示：

```
transition: property duration timing-function delay;
```

可以看到，transition是一个简写的属性，它可以用来设置4个过渡的属性，这4个属性的描述分别如下。

（1）transition-property：规定设置过渡效果的CSS属性的名称。

（2）transition-duration：规定设置过渡效果需要的时间，单位可以是秒（s）或者毫秒（ms）。

（3）transition-timing-function：规定速度效果的速度曲线。

（4）transition-delay：定义过渡效果何时开始。

例如下面的代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>8.1</title>
<style type="text/css">
```

```
a{
padding:5px 20px;
text-align:center;
text-decoration:none;
font:bold 25px Arial;
color:white;
background-color:#3e5706;
transition:all 300ms ease;
border-radius: 30px;
}
a:hover{
transform: rotateY(360deg);
transition-delay:0.2s;
color:#19667d;
background-color:#70c9e3;
}
</style>
</head>
<body>
<a href="#">button</a>
</body>
</html>
```

在这段代码中，页面只有一个超链接标签，我们通过CSS将其设置成一个可以翻转的按钮。CSS中只有两个选择器，在a选择器中，首先设置a元素的上下内边距为5个像素，左右内边距为20个像素，取消超链接默认的下划线设置，定义超链接的字体和大小，文本颜色设置为白色，然后为其设置一个背景色。这里关键的一段代码如下：

```
transition:all 300ms ease;
```

这里的all表示a元素所有的属性；300ms表示300毫秒；ease表示以慢速开始，然后变快，最后以慢速结束的过渡效果。最后一句代码“border-radius: 30px”表示超链接边框的圆角半径为30个像素，至此就设置了一个圆角按钮的效果，效果如图8.7所示。

另一个选择器a:hover设置当鼠标悬停在超链接上时超链接的效果。首先使用transform属性设置超链接沿着Y轴旋转360度，然后再使用transition-delay属性设置延时0.2秒，同时设置超链接文本的颜色和背景色。刷新页面，当鼠标停留在按钮上时，按钮会先翻转，然后变换为另一种样式，效果如图8.8所示。

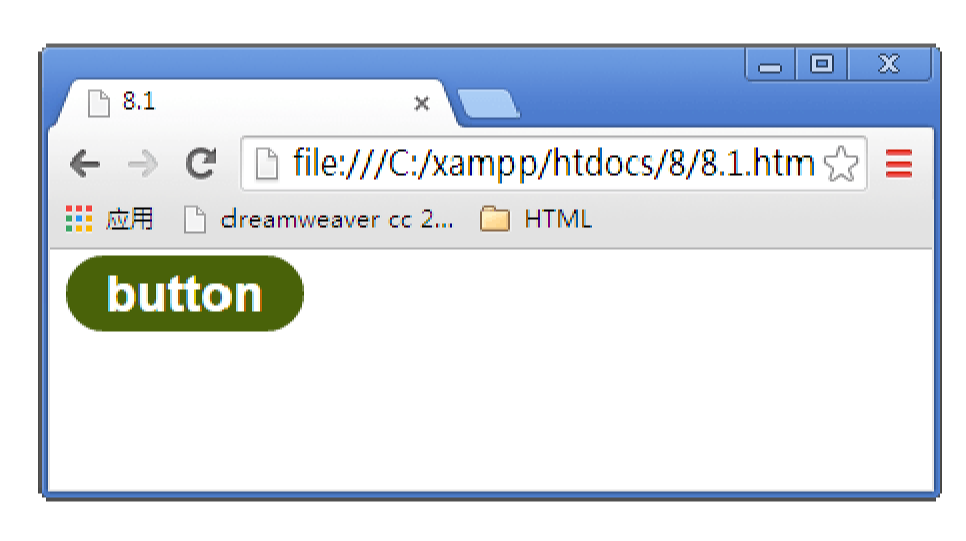



图8.7

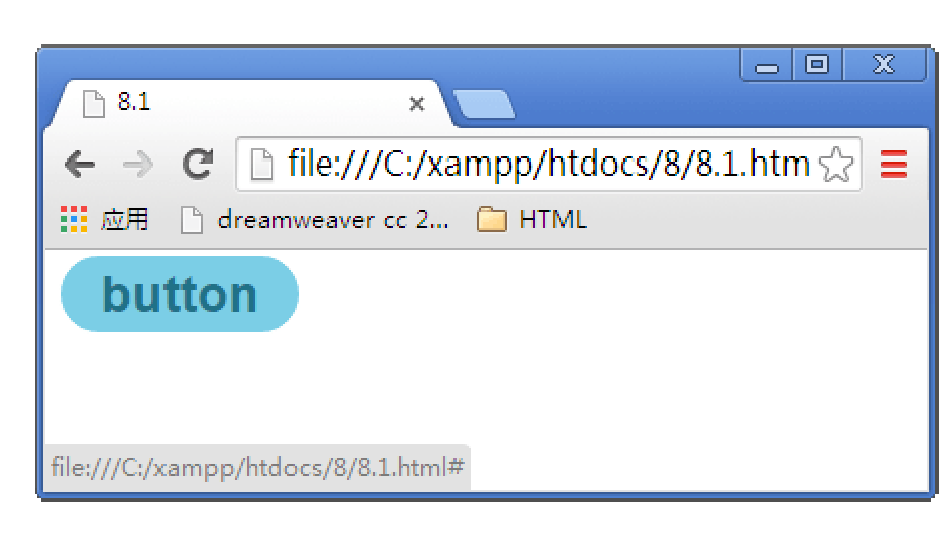


图8.8

8.3.2 在Dreamweaver中制作翻转按钮

翻转按钮效果是目前使用比较广泛的一种按钮效果，它可以有效地美化网页。除了以上介绍的使用CSS实现翻转按钮效果外，我们还可以使用Dreamweaver轻松制作翻转按钮效果，具体操作步骤如下：

01 打开Dreamweaver软件，新建一个HTML文件。

02 选择“插入”→“图像”→“鼠标经过图像”命令，如图8.9所示。

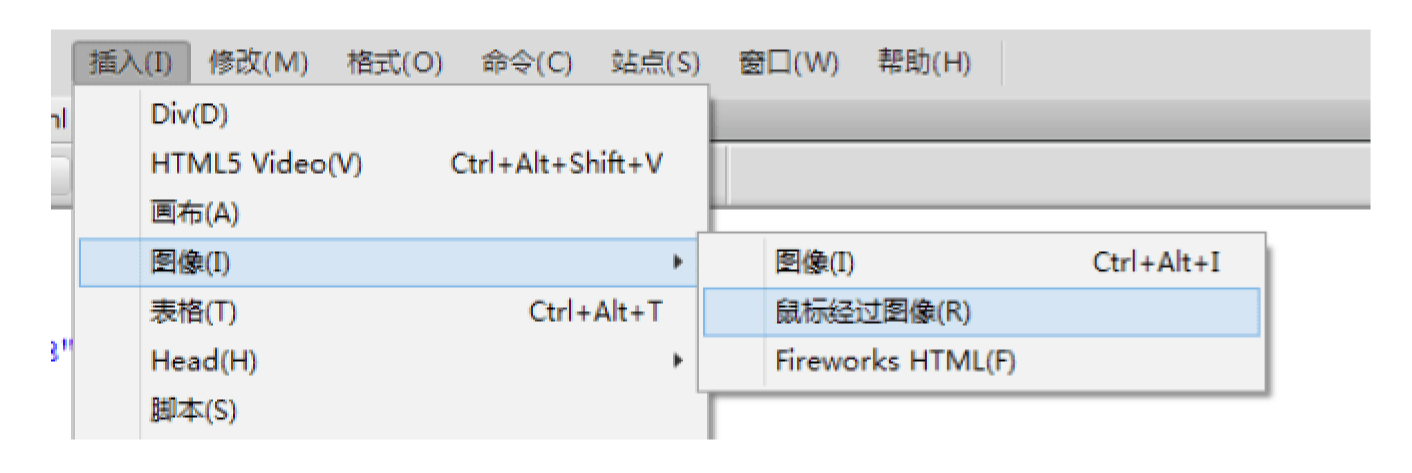


图8.9

03 打开“插入鼠标经过图像”对话框，如图8.10所示。在这个对话框中，设置“图像名称”为“button”，单击“原始图像”后面的“浏览”按钮，打开“选择文件”对话框，选择原始的按钮图像，然后返回“插入鼠标经过图像”对话框。单击“鼠标经过图像”后面的“浏览”按钮，设置鼠标经过时要显示的图像；设置替换文本为button；设置当按下按钮时，前往的URL为#。单击“确定”按钮后，完成了一个翻转按钮的效果。

插入鼠标经过图像
图像名称：button
原始图像：img/img01.png 浏览...
鼠标经过图像：img/img02.png 浏览...
☑ 预载鼠标经过图像
替换文本：button
按下时，前往的 URL：# 浏览...
确定
取消
帮助

图8.10

04 经过以上操作后，在Dreamweaver的代码视图中，我们可以看到页面中添加了很多代码，包括JavaScript、HTML元素和属性，完整代码如下：

```
    <!doctype html>
    <html>
    <head>
    <meta charset="utf-8">
    <title>8.2</title>
    <script type="text/javascript">
    function MM_swapImgRestore() { //v3.0
       var i,x,a=document.MM_sr; for(i=0;a&&i<a.length&&(x=a[i])&&x.oSrc;i++)
x.src=x.oSrc;
    }
```

```
    function MM_preloadImages() { //v3.0
      var d=document; if(d.images){ if(!d.MM_p) d.MM_p=new Array();
          var i,j=d.MM_p.length,a=MM_preloadImages.arguments; for(i=0; i<a.
length; i++)
          if (a[i].indexOf("#")!=0){ d.MM_p[j]=new Image; d.MM_p[j++].
src=a[i];}}
    }
    function MM_findObj(n, d) { //v4.01
      var p,i,x;  if(!d) d=document; if((p=n.indexOf("?"))>0&&parent.frames.
length) {
        d=parent.frames[n.substring(p+1)].document; n=n.substring(0,p);}
      if(!(x=d[n])&&d.all) x=d.all[n]; for (i=0;!x&&i<d.forms.length;i++) x=d.
forms[i][n];
       for(i=0;!x&&d.layers&&i<d.layers.length;i++) x=MM_findObj(n,d.layers[i].
document);
      if(!x && d.getElementById) x=d.getElementById(n); return x;
    }
    function MM_swapImage() { //v3.0
       var i,j=0,x,a=MM_swapImage.arguments; document.MM_sr=new Array;
for(i=0;i<(a.length-2);i+=3)
        if ((x=MM_findObj(a[i]))!=null){document.MM_sr[j++]=x; if(!x.oSrc)
x.oSrc=x.src; x.src=a[i+2];}
    }
    </script>
    </head>
    <body onLoad="MM_preloadImages('img/img02.png')">
    <a href="#" onMouseOut="MM_swapImgRestore()" onMouseOver="MM_
swapImage('button','','img/img02.png',1)"><img src="img/img01.png"
alt="button" width="104" height="33" id="button"></a>
    </body>
    </html>
```

05 与CSS实现翻转效果相比，这些代码显得过于冗长和繁琐，同时也不利于维护和更新，所以我们不建议这样操作。

06 刷新浏览器，可以看到如图8.11所示按钮图像，当鼠标移动到按钮上时，可以看到如图8.12所示按钮图像。

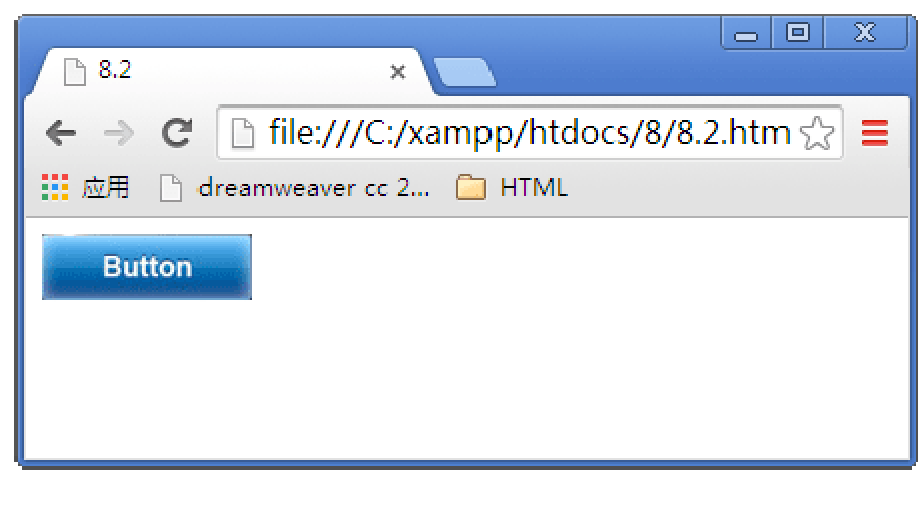


图8.11

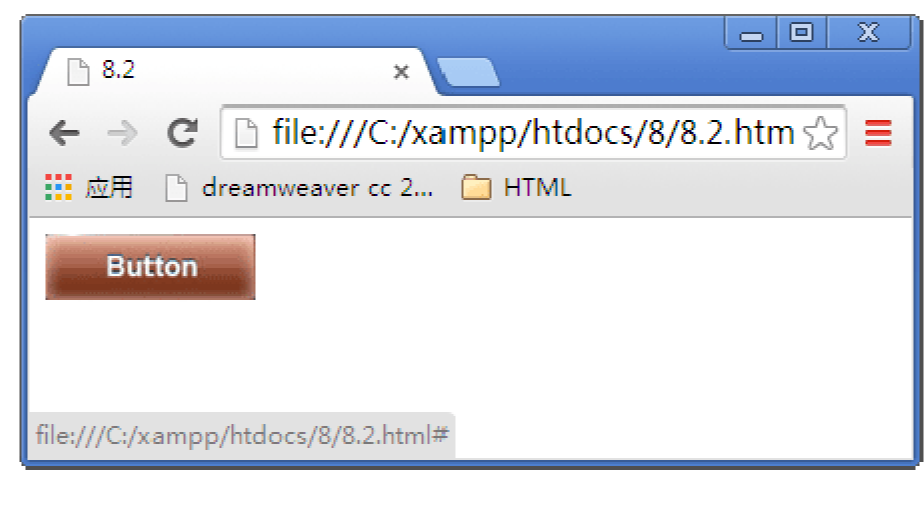


图8.12



8.4 用CSS创建导航菜单

如今，大部分网页都使用CSS和列表创建导航菜单，这是因为较之JavaScript和图片的创建方式，这种方式创建的导航菜单节省了很多的代码，而且读取速度更快。不仅如此，使用CSS来设计导航菜单并不会限制个人的想法，各种灵活多变的设计方案呈现出了不同的视觉效果。

8.4.1 创建CSS列表导航菜单

我们先使用CSS和列表创建一个简单的导航菜单，详细操作步骤如下：

01 新建一个HTML文件，在页面中创建一个列表，每一个列表项中都有一个超链接，详细代码如下：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>8.3</title>
</head>
<body>
<ul class="nav">
  <li class="home"><a href="#">home</a></li>
  <li class="setting"><a href="#">setting</a></li>
  <li class="about"><a href="#">about</a></li>
</ul>
</body>
</html>
```

运行这段代码后，效果如图8.13所示。

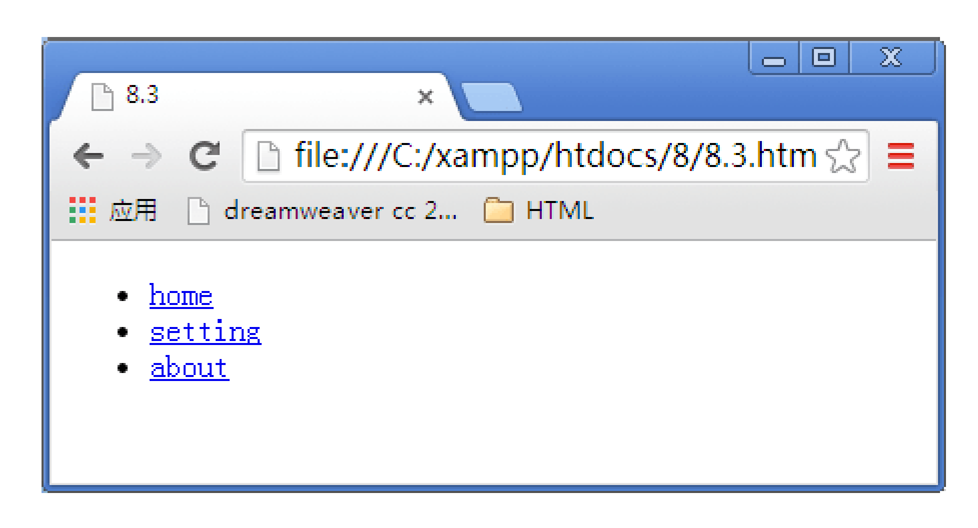


图8.13

02 先设置列表的宽度和高度，然后设置内边距和边框，最后设置一个背景图像，注意这里使用的背景图像只有一个像素宽度，所以需要使用repeat-x属性指定在X轴上重复，相关代码如下：

```
.nav {
width: 500px;
height: 50px;
padding: 5px 15px;
background:url(img/bg.png) repeat-x;
border: 1px solid #efefef;
}
```

刷新页面后，效果如图8.14所示。

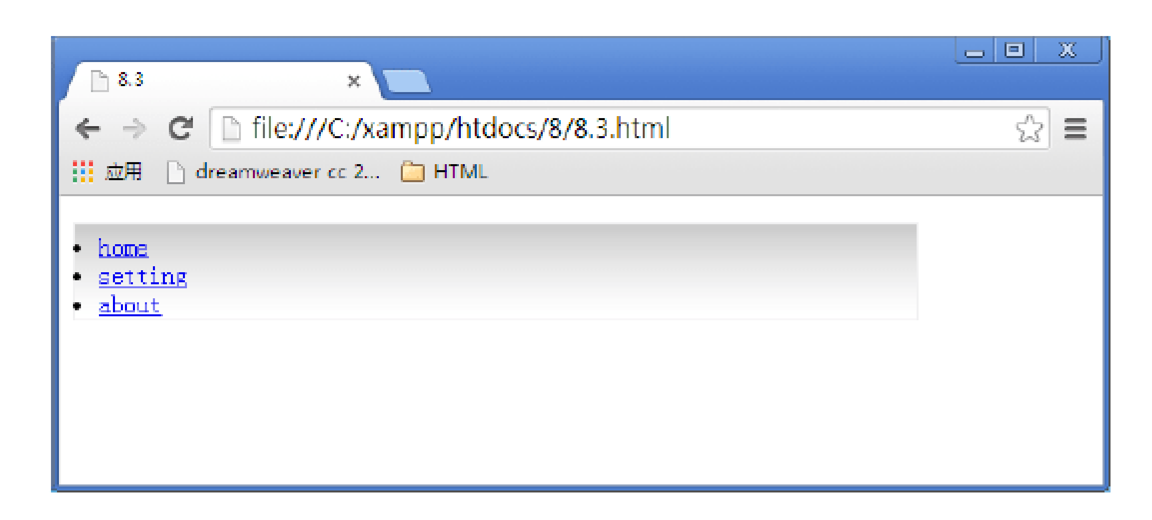


图8.14

03 设置li标签的样式。首先使用float属性让所有的列表项向左浮动，然后使用display属性让其显示在一行，最后设置列表项的尺寸和外边距，相关代码如下：

```
.nav li {
float: left;
width: 125px;
height: 50px;
display: inline;
margin-right:25px;
}
```

刷新页面后，效果如图8.15所示。

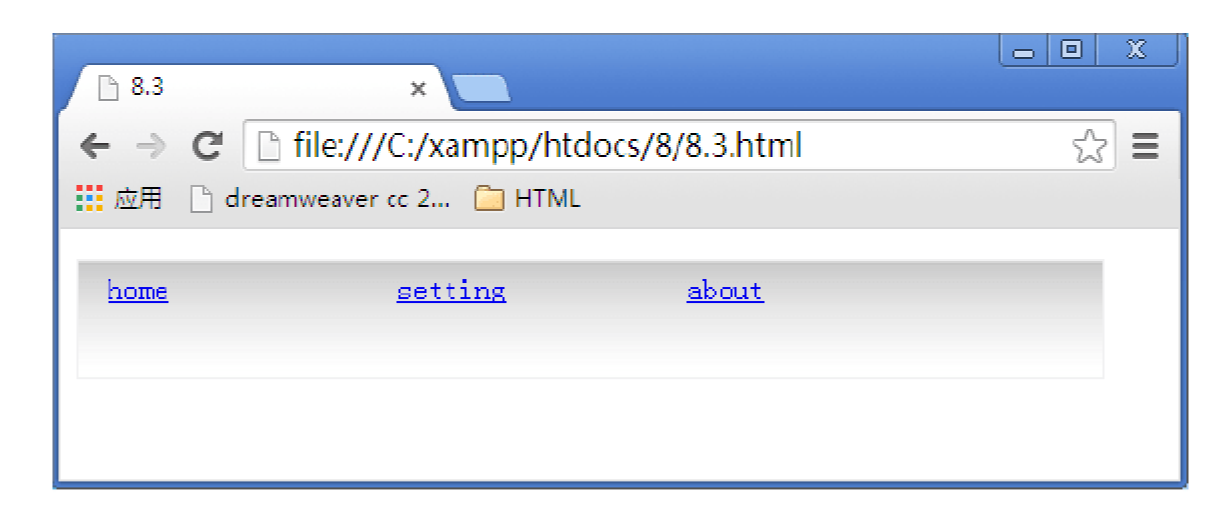


图8.15

04 设置列表项中的超链接，使用display属性将超链接转换成块元素，设置内边距、颜色、字体、大小和高度，并取消超链接文本的下划线。然后设置当鼠标停留在超链接上时，改变超链接的颜色，相关代码如下：

```
.nav li a {
display: block;
padding: 15px 0px 0px 50px;
color: #000;
font-size: 18px;
```

```
font-family: arial;
height: 35px;
text-decoration: none;
}
.nav li a:hover {
color: #C00;
}
```

刷新页面后，将鼠标停留在超链接上，效果如图8.16所示。

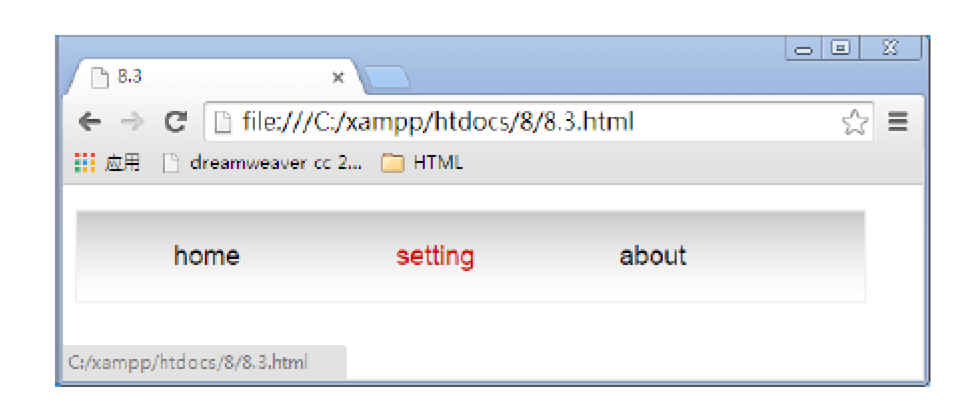


图8.16

05 在上一步操作中，我们已经使用padding属性在超链接左边空出了50个像素的位置，下面用background属性为菜单添加背景图像，并设置no-repeat属性，同时还需要为鼠标悬停设置另一种背景图像，相关代码如下：

```
li.home {
background: url(img/home.png) no-repeat;
}
li.home:hover {
background: url(img/home-2.png) no-repeat;
}
li.about {
background: url(img/key.png) no-repeat;
}
li.about:hover {
background: url(img/key-2.png) no-repeat;
}
li.setting {
background: url(img/settings.png) no-repeat;
}
li.setting:hover {
background: url(img/settings-2.png) no-repeat;
}
```

刷新页面后，当鼠标悬停在超链接上时，背景图像也会发生变化，效果如图8.17所示。

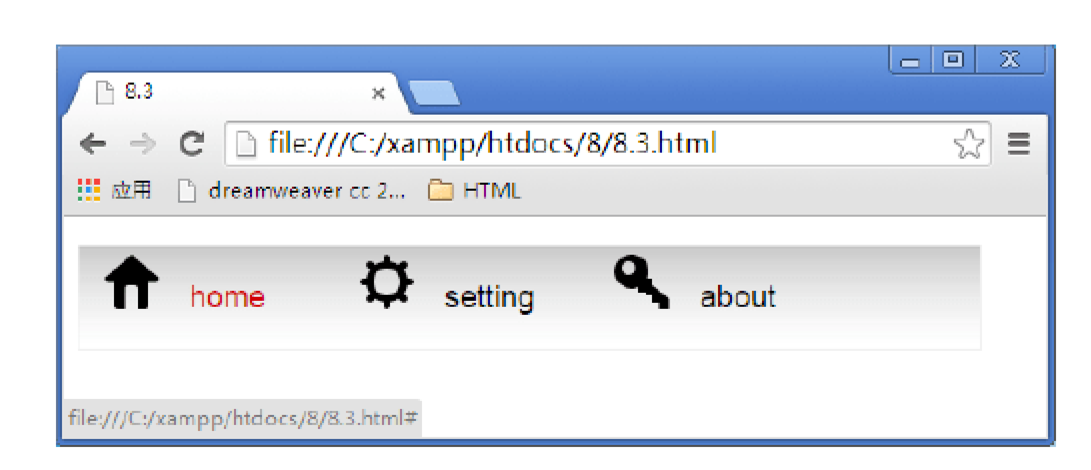


图8.17

8.4.2 创建二级CSS列表导航菜单

为了让导航菜单能够包含更多的页面检索内容，通常情况会使用二级甚至三级导航菜单。使用二级导航菜单不仅可以满足我们的要求，而且还符合人们对于导航菜单的习惯操作，如果使用三级菜单，有可能会让某些用户感觉不适应，从而放弃更多内容的检索。

下面我们就详细介绍如何使用CSS列表创建二级导航菜单，操作步骤如下：

01 新建一个HTML页面，在页面中创建两个列表，并将一个列表嵌套在另一个列表中，详细代码如下：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>8.4</title>
</head>
<body>
<div id="nav">
  <ul>
    <li><a href="#">主页</a></li>
    <li><a href="#">解决方案</a></li>
    <li><a href="#">服务</a>
      <ul>
        <li><a href="#">咨询服务</a></li>
        <li><a href="#">管理服务</a></li>
        <li><a href="#">客户支持</a></li>
      </ul>
    </li>
    <li><a href="#">产品</a>
      <ul>
        <li><a href="#">互联网业务</a></li>
        <li><a href="#">智能设备</a></li>
        <li><a href="#">云计算</a></li>
      </ul>
    </li>
    <li><a href="#">技术支持</a></li>
  </ul>
</div>
</body>
</html>
```

运行这段代码后，效果如图8.18所示。

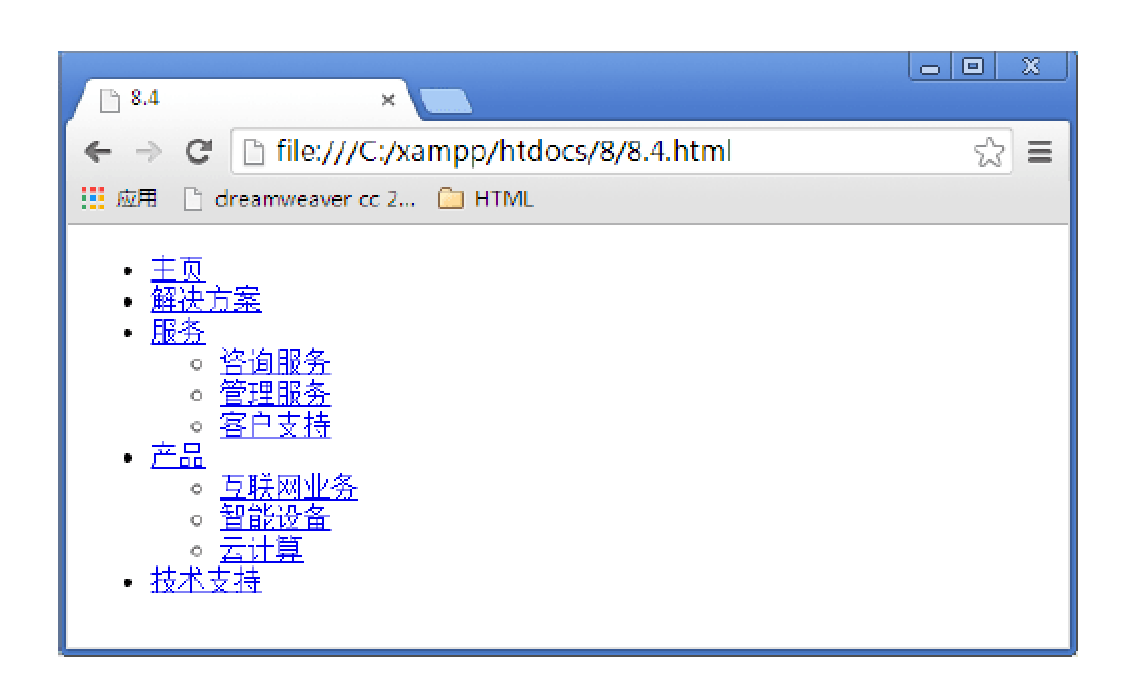


图8.18

02 下面我们依次为导航菜单添加样式。首先清除所有元素的内边距和外边距，然后为div元素设置样式，设置其宽度为800px，高度为50px，div中所有字体为粗体，字号为26px，字体为arial。这里还会用到使用CSS控制元素居中的常用方法，设置margin左右边距为auto。相关代码如下：

```
* {
margin: 0;
padding: 0;
}
#nav {
width: 800px;
height: 50px;
font: bold 26px arial;
margin: 0 auto;
}
```

刷新页面后，效果如图8.19所示。

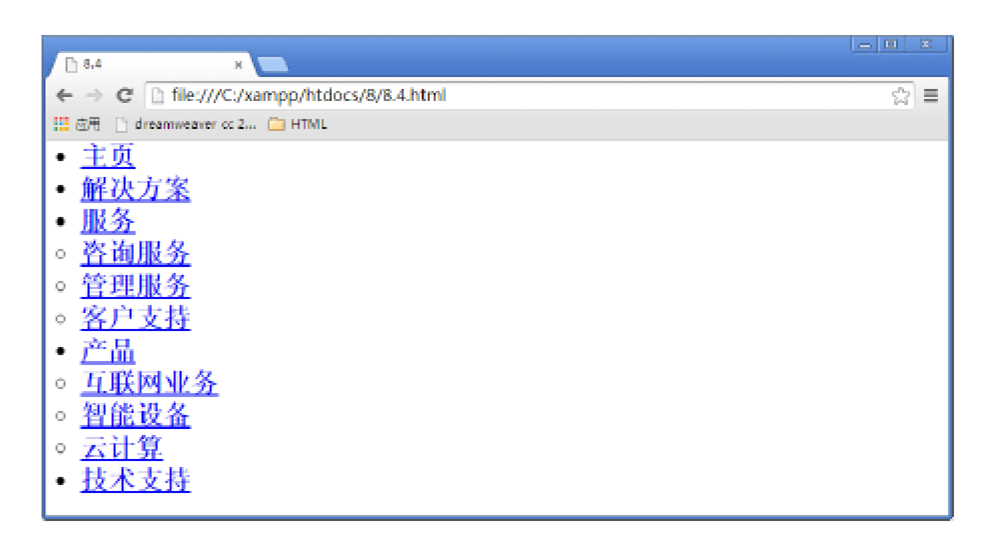


图8.19

03 外层列表作为一级菜单，首先需要清除列表符号，然后让这些列表项向左浮动，这样他们就能显示在一行，最后设置一级菜单的position属性为relative，这是为调整二级菜单做的准备。相关代码如下：

```
#nav ul li {
list-style: none;
float: left;
position: relative;
}
```

刷新页面后，效果如图8.20所示。



图8.20

04 下面设置一级菜单中超链接的样式。首先设置超链接的文本颜色为白色，然后添加一个背景色，去除超链接默认的下划线效果，让超链接中的文本居中显示。为了让菜单显示的更好看些，设置超链接内边距为20px，给文本和边框留一定的距离，将超链接转换成块级元素，并设置外边距为1px，让一级菜单之间留有空隙。相关代码如下：

```
#nav ul li a {
color: white;
background-color: #232E2E;
text-decoration: none;
text-align: center;
padding: 20px;
display: block;
margin-right: 1px;
}
```

刷新页面后，效果如图8.21所示。



图8.21

05 当鼠标悬停在一级菜单上时，设置一级菜单的文本颜色和背景色，然后使用display属性让二级菜单隐藏，相关代码如下：

```
#nav ul li:hover a {
color: white;
background-color: #AC9072;
}
#nav ul li ul {
display: none;
}
```

刷新页面后，当鼠标悬停在一级菜单上时，效果如图8.22所示。



图8.22

06 当鼠标悬停在一级菜单上时，设置display属性显示二级菜单，并设置二级菜单position属性为absolute，这是一个重要的设置。因为相对于一级菜单，让二级菜单绝对定位，才能在一级菜单下面正确显示二级菜单。相关代码如下：

```
#nav ul li:hover ul {
display: block;
position: absolute;
}
```

刷新页面后，当鼠标悬停在一级菜单上时，效果如图8.23所示。



图8.23

07 目前二级菜单显示的过于紧凑，为了改善这一效果，我们需要对二级菜单的超链接样式进行设置。首先将超链接转换成块元素，为其重新设置背景颜色，并设置一定的宽度，然后设置二级菜单的底边框为样式，这样二级菜单各菜单项之间也会有一个像素的间隙，相关代码如下：

```
#nav ul li:hover ul li a {
display: block;
background-color: #232E2E;
width: 150px;
border-bottom: solid 1px white;
}
```

刷新页面后，当鼠标移动到一级菜单上时，效果如图8.24所示。



图8.24

08 最后，为了让二级菜单和一级菜单具有类似的悬停效果，可以设置二级菜单超链接的悬停样式，改变文本颜色和背景色，相关代码如下：

```
#nav ul li:hover ul li a:hover {
color: #E67428;
background-color: #316893;
}
```

刷新页面后，当鼠标悬停在二级菜单上时，效果如图8.25所示。



图8.25

09 至此，用CSS和列表创建的二级菜单效果就完成了。

第9章 制作表单

HTML表单作为Web页面与用户交互的重要窗口，一直是网站开发过程中不可缺少的内容。HTML 5在Form表单的应用上提供了更强大的功能，本章主要介绍在HTML 5中表单的使用方法。



表单标签

HTML 5中新增了很多新的表单标签，结合原有的表单控件，方便程序设计者创建更复杂的应用，下面我们对这些表单标签进行详细介绍。

9.1.1 <form>标签

<form>标签用于创建一个表单，开始标签和结束标签之间的所有内容都是表单中的内容，当用户提交表单时，这些内容也会同时提交。<form>表单有很多属性，各属性的简单介绍如下表所示。

表9.1

属性	描述
action	定义一个URL。当单击提交按钮时，向这个URL发送数据
data	自动插入数据
replace	定义表单提交时做的事情
accept	处理表单的服务器可以正确处理的内容类型列表
accept-charset	表单数据的可能字符集列表（默认值是unknown）
enctype	用于对表单内容进行编码的MIME类型
method	用于向action URL发送数据的HTTP方法（默认是get）
target	在何处打开目标URL

其中常用的几个属性介绍如下。

（1）action属性：这个属性值可以是程序或脚本的一个完整的URL，也可以是表单要提交的地址。要提交的地址可以是绝对的，也可以是相对的，还可以是一些其他形式的地址，

例如电子邮箱地址。

```
<form action="mailto:zhangsan@163.com"> </form>
```

（2）method属性：这个属性值有两个，一个get，一个post。如果使用get，那么来访者输入的数据会附加在URL之后，由用户端直接发送至服务器，所以速度上会比post快，但缺点是数据长度受限，这也是method属性的默认值。如果使用post，表单数据将与URL分开发送，用户端的计算机会通知服务器来读取数据，所以没有数据长度上的限制，缺点是速度上会比post慢。另外，使用get会在发送的URL地址后面以明文的方式附加要发送的数据，而post将以密文的方式发送数据。

（3）enctype属性：如果指定属性值为text/plain，将以纯文本的形式传送；如果指定属性值为application/x-www-form-urlencoded，将以默认的编码传送；如果指定属性值为multipart/form-data MIME编码，上传文件的表单必须选择该项。

（4）target属性：target属性可供选择的值有4个，_blank是指将返回的信息显示在新打开的窗口中，_parent是指将返回的信息显示在父级的浏览器窗口中，_self则表示将返回的信息显示在当前浏览器窗口，_top表示将返回的信息显示在顶级浏览器窗口中。

使用<form>标签创建一个简单的表单界面，相关代码如下。运行这段代码后，效果如图9.1所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>9.1</title>
</head>
<body>
<form id="form" action="#" method="post">
  <label for="name">姓    名：</label>
  <input id="name" name="name" type="text" />  <br>
  <label for="address">居住地址：</label>
  <input id="address" name="address" type="text" />  <br>
  <label for="number">联系电话：</label>
  <input id="number" name="number" type="number" />  <br>
  <label for="email">电子邮箱：</label>
  <input id="email" name="email" type="email" />  <br>
  <input type="submit" value="提交" id="submit" name="submit" />
  <input type="reset" value="重置" id="reset" name="reset" />
</form>
</body>
</html>
```

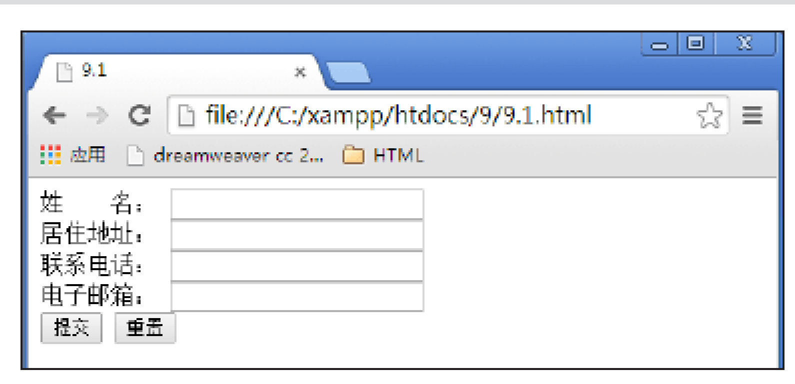


图9.1

9.1.2 <fieldset>标签

当表单中的内容很多时，可以使用<fieldset>标签将相关的内容进行分组，分组后的标签在浏览器中将以特殊的效果显示。例如下面这段代码，将两个输入标签放到<fieldset>标签中，运行这段代码后，效果如图9.2所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>9.2</title>
</head>
<body>
<form id="login" method="post" action="#">
<fieldset>
  <label for="username">用户名：</label>
  <input id="username" name="username" type="text" /> <br>
  <label for="password">密  码：</label>
  <input id="password" name="password" type="password" />
</fieldset>
</form>
</body>
</html>
```

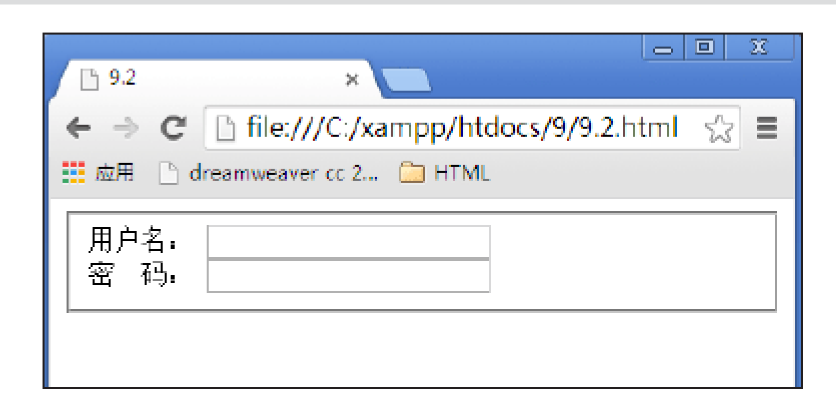


图9.2

9.1.3 <legend>标签

<legend>标签可以看作是一个标题标签，它可以为<fieldset>标签、<figure>标签以及<details>标签定义标题。例如，在上面的<fieldset>标签中添加下面这段代码：

```
<legend>登录</legend>
```

刷新页面后，效果如图9.3所示。

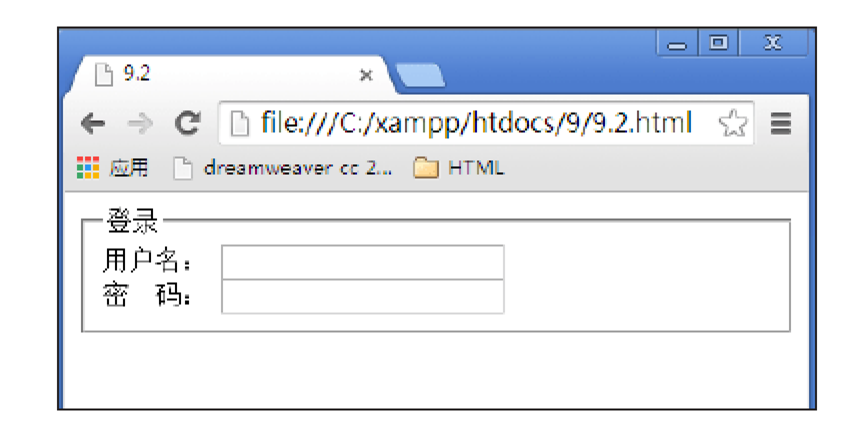


图9.3

9.1.4 <input>标签

<input>标签用于定义form表单中的输入字段，根据类型的不同，可以定义文本框、密码框、单选框、复选框等多种控件。其属性和可选值，以及相关描述如表9.2所示：

表9.2

属性	值	描述
accept	list_of_mime_types	一个逗号分隔的MIME类型列表，指示文件传输的MIME类型。只能与type="file"配合使用
alt	text	定义图像的替代文本。只能与type="image"配合使用
autocomplete		自动完成
autofocus	bool	当页面加载时，使输入字段获得焦点。当type="hidden"时无法使用。
checked	bool	指示此input元素首次加载时应该被选中
disabled	bool	当input元素首次加载时禁用此元素，这样用户就无法在其中写文本，或选定它。注意不能与type="hidden"一起使用
form	bool	定义输入字段属于一个或多个表单
inputmode	inputmode	定义预期的输入类型
list	id of a datalist	引用datalist元素。如果定义，则一个下拉列表可用于向输入字段插入值
max	number	输入字段的最大值
maxlength	number	定义文本域中所允许的字符的最大数目
min	number	输入字段的最小值
name	field_name	为input元素定义唯一的名称
readonly	readonly	指示是否可修改该字段的值
replace	text	定义当表单提交时如何处理该输入字段
required	bool	定义输入字段的值是否是必须的。当使用下列类型时无法使用：hiddden、image、buttom、submit、reset
src	URL	定义要显示的图像的URL，只有当type="image"时使用
template	template	定义一个或多个模板
type	button、checkbox、date、datetime、datetime-local、email、file、hidden、image、month、number、password、radio、range、reset、submit、text、time、url、week	指示input元素的类型。默认值是"text"，表示文本字段，password表示密码域，radio表示单选按钮，checkbox表示复选框，button表示普通按钮，submit表示提交按钮。提交按钮是一种特殊的按钮，不需要设置onclick参数，在单击该类型按钮时可以实现表单内容的提交。reset表示重置按钮，单击该按钮时清除用户在页面上的输入信息。image表示图像域，hidden表示隐藏域，file表示文件域。
value	value	对于按钮、重置按钮和确认按钮：定义按钮上的文本。对于图像按钮：定义传递向某个脚本的此域的符号结果。对于复选框和单选按钮：定义input元素被点击时的结果。对于隐藏域、密码域以及文本域：定义元素的默认值

例如下面这段代码，在表单中定义多种类型的输入控件。运行这段代码后，效果如图9.4所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>9.3</title>
</head>
<body>
<form id="form" action="#" method="post">
  <fieldset>
    <legend>基础信息</legend>
    <label for="username">用户名：</label>
    <input type="text" id="username" />    <br>
    <label for="password">密  码：</label>
    <input type="password" id="password" />    <br>
    <label>性  别：</label>
    <input type="radio" name="sex" id="sex" value="1" />
    <label for="sex">男</label>
    <input type="radio" name="sex" id="sex" value="2" />
    <label for="sex">女</label>    <br>
    <label for="address">地  址：</label>
    <input type="text" id="address" />    <br>
    <label for="number">电  话：</label>
    <input type="number" id="number" />    <br>
  </fieldset>
  <fieldset>
    <legend>推送消息</legend>
    <label for="fav">推荐新闻</label>
    <input type="checkbox" id="fav" name="fav" value="1"/>
    <label for="fav">娱乐</label>
    <input type="checkbox" id="fav" name="fav" value="2"/>
    <label for="fav">体育</label>
    <input type="checkbox" id="fav" name="fav" value="3"/>
    <label for="fav">视频</label>
    <input type="checkbox" id="fav" name="fav" value="4"/>
  </fieldset>
  <fieldset>
    <legend>添加附件</legend>
    <label for="img">图像</label>
    <input type="file" id="img" name="img" size="50" maxlength="255"/>
  </fieldset>
  <fieldset>
    <legend>提交</legend>
    <input type="submit" value="submit" id="submit" name="submit"/>
    <input type="reset" value="reset" id="reset" name="reset"/>
  </fieldset>
</form>
</body>
```

```
</html>
```

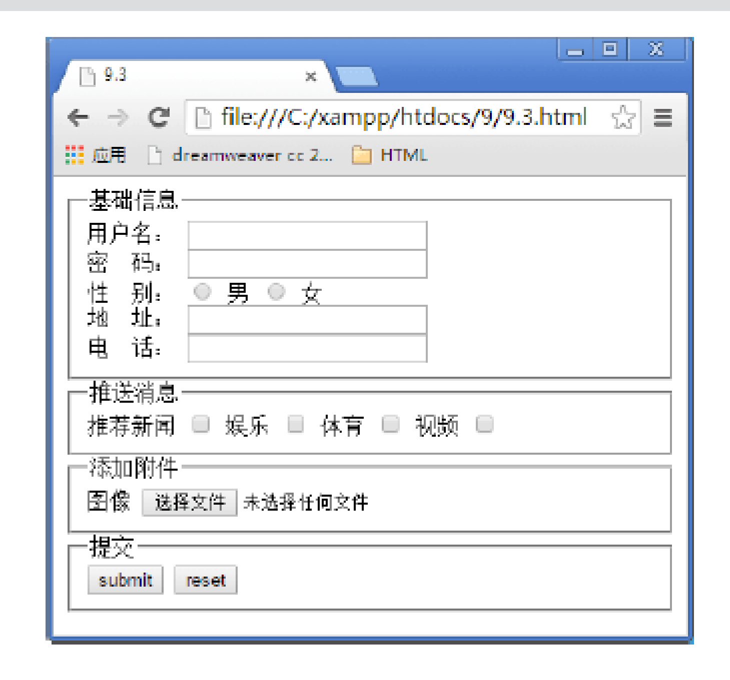


图9.4

9.1.5 <select>标签

<select>标签用于创建下拉列表，常用的属性介绍如下表9.3所示。

表9.3

属性	值	描述
autofocus	bool	在页面加载时使用这个select字段获得焦点
data	url	供自动插入数据
disabled	bool	当该属性为true时，会禁用该菜单
form	bool	定义select字段所属的一个或多个表单
multiple	bool	当该属性为true时，规定可一次选定多个项目
name	unipue_name	定义下拉列表的唯一标识符

例如，在form表单中<select>标签创建性别选择功能，相关代码如下所示：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>9.4</title>
</head>
<body>
<form id="form" action="#" method="post">
  <fieldset>
    <legend>性别</legend>
    <select multiple="false" id="sex" name="sex">
      <option>男</option>
      <option>女</option>
      <option>保密</option>
    </select>
  </fieldset>
</form>
</body>
</html>
```

运行这段代码后，在页面中可以看到如图9.5所示的下拉列表。

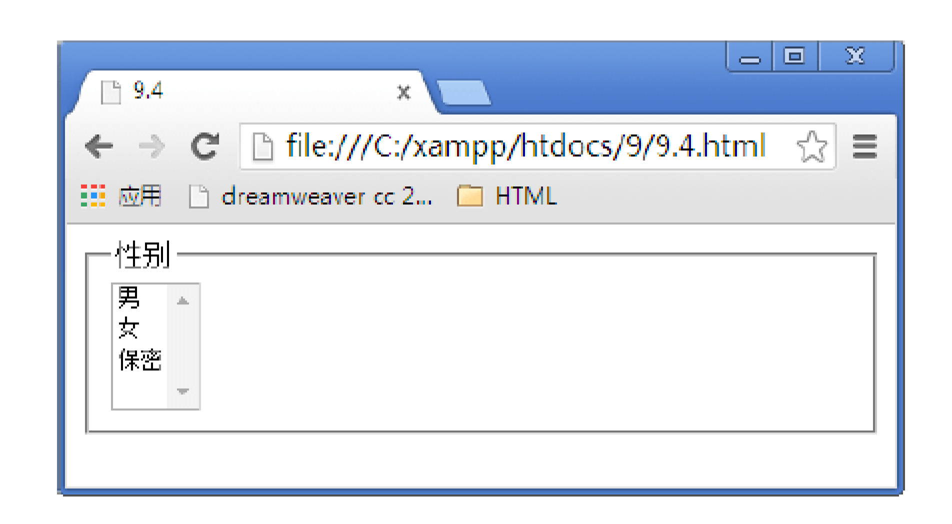


图9.5

9.1.6 <option>标签

<option>标签用于为<select>标签或<datalist>标签添加选项，每一个<option>标签代表一个选项。该标签有4个属性，disabled属性规定此选项应在首次加载时被禁用，label属性定义使用<optgroup>时所使用的标注，selected属性规定选项表现为选中状态，value属性定义送往服务器的选项值。需要注意的是，<option>标签在其他地方使用没有任何意义。

9.1.7 <optgroup>标签

<optgroup>标签用于定义一个选项组。在这个选项组中，还可以进一步对组中的选项进行划分。<optgroup>标签的label属性用于定义选项组的标注，disabled属性用于设置在首次加载时，禁用该选项组。例如下面的代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>9.5</title>
</head>
<body>
<form id="form" action="#" method="post">
  <fieldset>
    <legend>选择行业</legend>
    <select multiple="multiple" id="jobs" name="jobs">
      <optgroup label="生产制造">
      <option value="汽车制造">汽车制造</option>
      <option value="印刷包装">印刷包装</option>
      <option value="机械机床">机械机床</option>
      </optgroup>
      <optgroup label="服务业">
      <option value="运动健身">运动健身</option>
      <option value="酒店旅游">酒店旅游</option>
      <option value="美容保健">美容保健</option>
      <option value="公务员">公务员</option>
      </optgroup>
```

```
      </select>
    </fieldset>
  </form>
  </body>
  </html>
```

运行这段代码后，效果如图9.6所示。

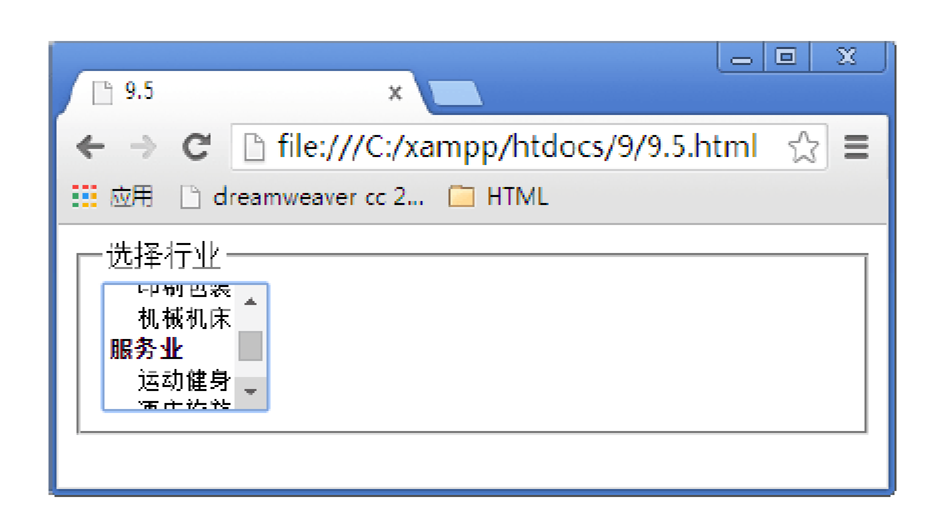


图9.6

9.1.8 <textarea>标签

<textarea>标签用于定义一个多行文本输入区域，用户可以在此文本区域中输入文本，文本的数量没有限制。<textarea>标签有很多有用的属性，如表9.4所示。

表9.4

属性	值	描述
autofocus	bool	在页面加载时该textare获得焦点
cols	number	规定文本区域可见的列数
disabled	bool	当次文本区首次加载时禁用此文本
form	bool	定义该textarea所属的一个或多个表单
inputmode	inputmode	定义该textarea所期望的输入类型
name	name_of_textarea	为此文本区规定一个名称
readonly	bool	指示用户无法修改文本区内的内容
required	bool	定义为了提交表单，该textarea的值是否是必须的
rows	number	规定文本区内可见的行数

例如下面这段代码，使用textarea标签创建了一个留言框，运行效果如图9.7所示。

```
  <!doctype html>
  <html>
  <head>
  <meta charset="utf-8">
  <title>9.6</title>
  </head>
  <body>
  <form id="form" action="#" method="post">
    <fieldset>
      <legend>评论</legend>
```

```
        <label for="note">把你的想法说出来吧</label><br>
          <textarea id="note" name="note" cols="50" rows="10" autofocus></
textarea>
      </fieldset>
      <input type="submit" value="提交" id="submit" name="submit"/>
      <input type="reset" value="重置" id="reset" name="reset"/>
    </form>
    </body>
    </html>
```

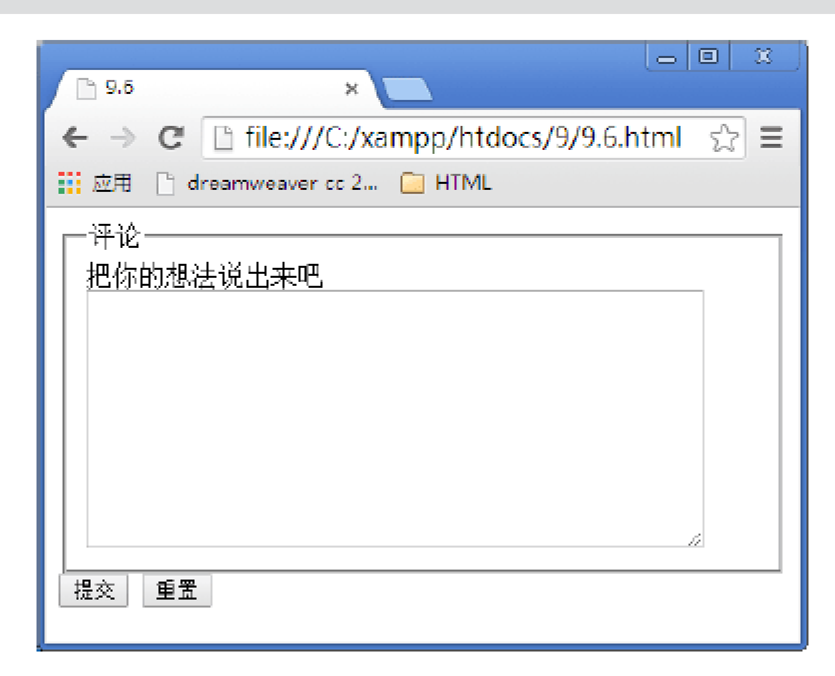


图9.7



9.2 创建表单结构

表单其实就是以<form>标签为容器，嵌套其他表单标签的一种结构。所有的表单标签必须嵌套在<form>标签内部才有效，表单标签的主要作用就是收集用户的输入信息。

最基本的表单至少要包含3个属性，id、action和method。id属性用于指定表单的唯一性；action属性指定表单的处理程序，当点击表单中的提交按钮时，表单中的所有数据将发送到action属性指定的服务器进行处理；method属性指定浏览器如何处理发送的数据，如果是post则以密文的方式传递数据，如果是get，则以明文的方式传递数据。

例如下面这段代码：

```
    <!doctype html>
    <html>
    <head>
    <meta charset="utf-8">
    <title>9.7</title>
    </head>
    <body>
    <form id="Myform" action="http://www.aa.com/do" method="post">
      <p>用户名：<input name="username" type="text" id="username"/></p>
      <p>密  码：<input name="userpass" type="password" id="userpass"/></p>
```

```
    <input type="submit" value="提交" id="submit" name="submit"/>
   <input type="reset" value="重置" id="reset" name="reset"/>
</form>
</body>
</html>
```

在这段代码中，指定表单的id为Myform，action属性指定为URL地址，最后的do表示提交数据的处理程序，而method指定提交的数据以post的方式提交。在这个表单中，前两个input标签用于输入用户名和密码，后两个input标签用户设置提交按钮和重置按钮。运行这段代码后，效果如图9.8所示，这就是一个最基本的表单结构。



图9.8



9.3 验证表单

表单是网站与用户交互的窗口，用来收集用户的输入信息，但有时恶意或无效的信息，会影响到网站的安全和正常运行。为了避免这些事情的发生，使用表单在收集信息的时候需要对数据进行有效的验证。

9.3.1 表单验证的原理

表单验证可以分为两类，一类是服务端验证，另一类是客户端验证。服务端验证需要通过一个服务端程序来执行验证操作，用户需要将数据提交到服务端，服务端根据相关程序验证用户提交的数据是否符合要求，如果不符合要求，则会抛出一个错误信息。网站用户登录验证就是服务端验证最典型的应用。

客户端验证与服务端验证最大的一个区别就是，客户端验证不需要用户将数据提交到服务端去验证，而是在用户提交数据之前，对这些数据进行验证，这样可以有效避免不必要的与服务端交互的过程。因为很多情况下，用户提交的数据在客户端就能完整有效性的验证，这些操作不需要去服务端执行。例如新用户注册时，输入密码的有效性验证，验证密码长度和复杂度是否符合要求，这类操作完全可以在客户端完成验证。

在HTML 5中，input标签的type属性可以指定用户输入内容的类型，这样就可以完成对这些数据的初步验证。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>9.8</title>
</head>
<body>
<form id="form" action="#" method="post">
<h2>电子邮箱</h2>
<input type="email" />
<input type="submit" value="提交"/>
</form>
</body>
</html>
```

在这段代码中，input标签的type类型为email，如果用户输入的信息不符合email的格式，那么在提交按钮时就会抛出一个错误提示。如图9.9所示是谷歌浏览中的提示效果，不同的浏览器提示信息的格式会有差异。

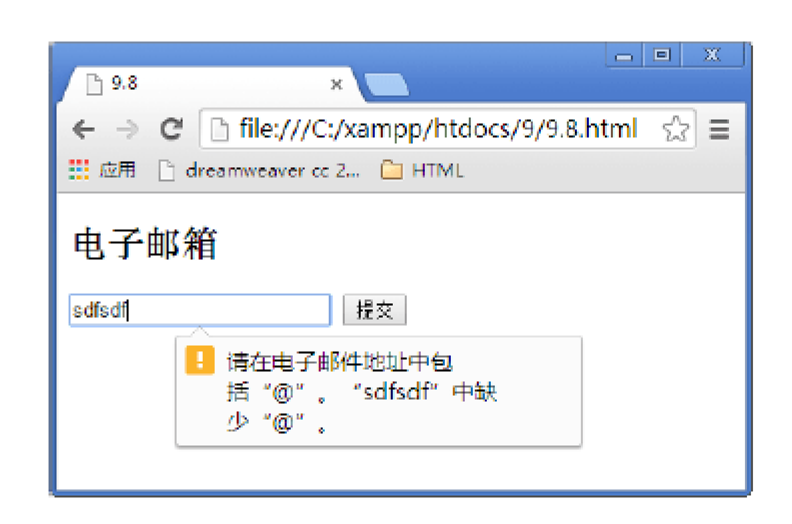


图9.9

input标签其他类型比如url、number、date等，都可以在提交数据之前完成有效性的验证。需要注意的是，通过HTML 5标签来完成数据验证的操作，只能在点击提交按钮的时候才能完成，而且HTML 5标签只提供了这些类型最基本的验证功能，如果需要更负责的完成验证功能，还需要配合JavaScript功能的使用。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>9.9</title>
<script language="javascript">
function check(){
var msg="";
var value=document.getElementById("email").value;
if(value==null || value=="")
{
msg="请输入电子邮箱地址！";
}
if(value.indexOf(‘@’)==-1)
{
msg="电子邮箱地址必须包含@";
```

```
    }
    document.getElementById("msg").innerHTML=msg;
    }
    </script>
    </head>
    <body>
    <form id="form" action="#" method="post">
    <h2>电子邮箱</h2>
    <input type="email" id="email" onBlur="check()" />
    <input type="submit" value="提交"/><br>
    <label id="msg"></label>
    </form>
    </body>
    </html>
```

在这段代码中，我们在input标签中添加了onBlur事件。当input标签失去焦点时触发check方法，check由JavaScript实现，先判断input控件的值是否为空，如果为空，那么设置提示信息“请输入电子邮箱地址！”，然后判断值是否包含@，如果不包含，那么设置提示信息“电子邮箱地址必须包含@”，最后将提示信息赋值到label标签。运行这段代码后，在输入框中随便输入几个字母，然后点击空白处，这样就会显示错误信息，效果如图9.10所示。

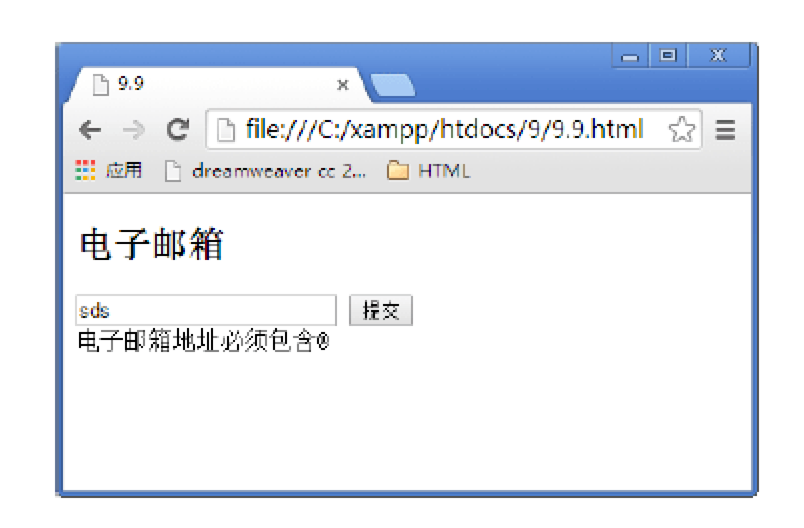


图9.10

HTML 5中还提供了required属性，当input标签添加该属性后，输入的内容不能为空，而pattern属性还可以指定输入内容必须符合指定的正则表达式，例如下面这段代码：

```
    <input id="phone " name="phone " type="text" required pattern="\d{3}-\
d{4}-\d{4}" placeholder="xxx-xxxx-xxxx"/>
```

这段代码表示一个输入文本框，这个文本框不能为空，而且输入的内容必须满足pattern指定的正则表达式。

9.3.2 在Dreamweaver中添加表单验证行为

在Dreamweaver中，还可以通过可视化操作界面为表单添加验证行为，详细操作步骤如下：

01 新建一个HTML文件，将光标定位到body元素内。

02 选择“插入”→“表单”→“表单”命令，在HTML页面内插入一个表单，然后以相同的方法再插入一个文本，如图9.1所示。



图9.11

03 将光标定位到插入的文本框中，可以在属性面板中看到文本框的相关属性。设置Name属性值为phone，勾选Auto Focus和Required两个复选框，表示该输入框自动获得焦点，而且是必填信息；设置PlaceHolder属性值为xxx-xxxx-xxxx，表示该文本框的提示信息；设置Pattern属性值为\d{3}-\d{4}-\d{4}，表示该文本框的内容必须符合这个正则表达式，效果如图9.12所示。

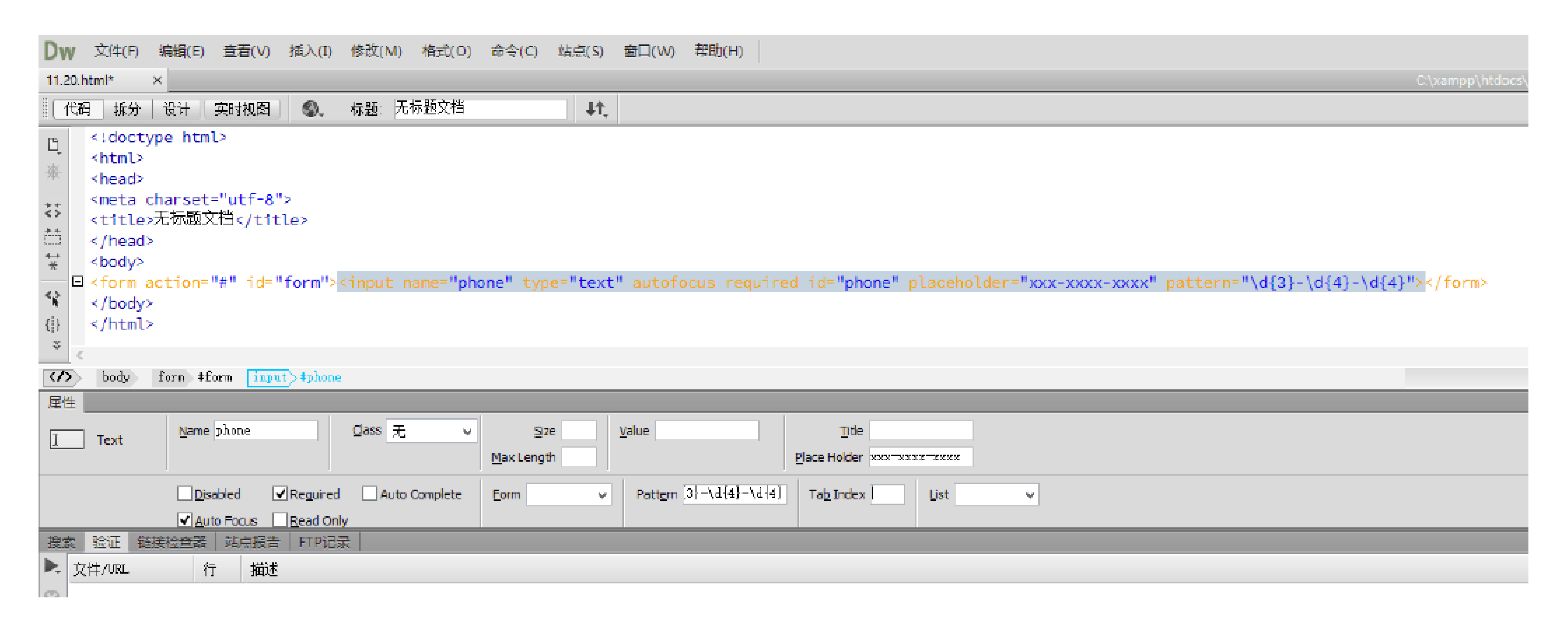


图9.12



使用在线表单服务

作为页面和用户交互的窗口，表单的设计越人性化就越能抓住用户的眼球。我们设计一

个表单需要花费很长的时间和精力，结果还可能令人不满意。使用在线表单服务，就可以直接用拖拽的方式，轻松创建各种漂亮的表单。

目前可使用的在线表单服务很多，基本上通过简单的注册后就可以使用，如图9.13所示的界面就是一个非常方便快捷的在线表单服务网站截图。在左侧的选择区域中，已经提供了各种常用表单的控件，点击相应的控件就可以将其添加到右边的预览窗口中，在预览窗口中选中添加的控件，左边的区域就会显示该控件的相关属性，如图9.14所示，用户可以根据自己的需要设置相关的属性。最终完成设置后，保存表单即可。

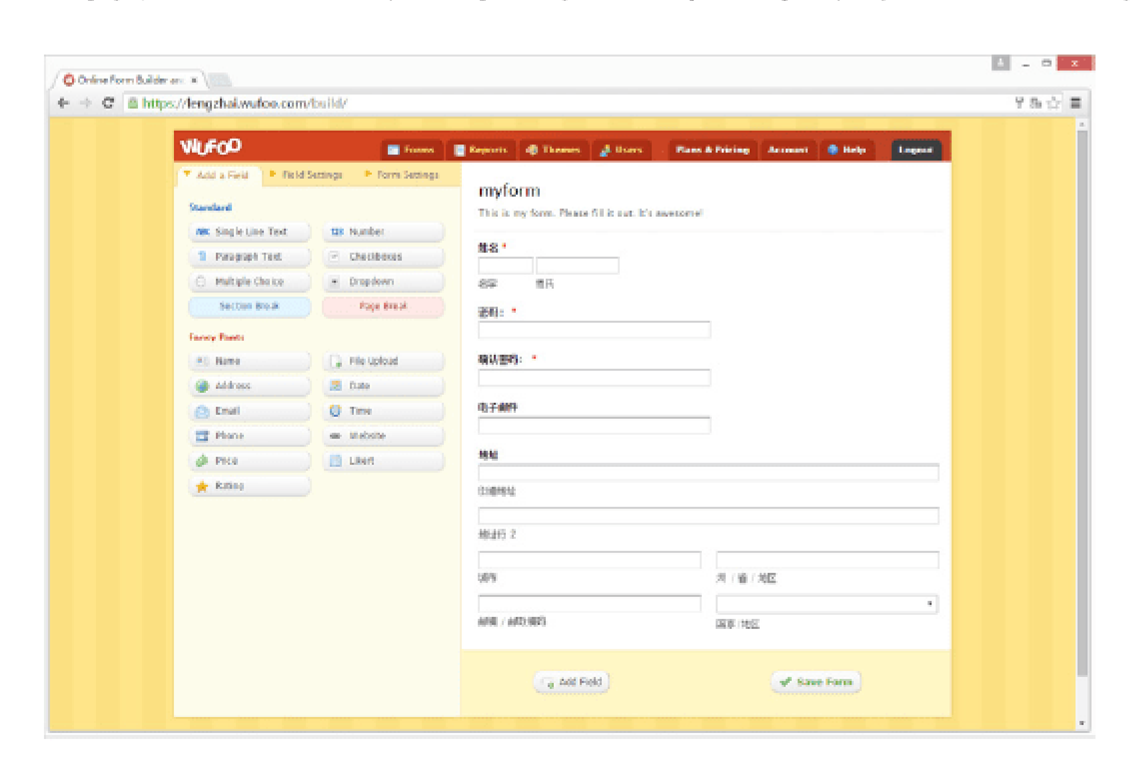


图9.13

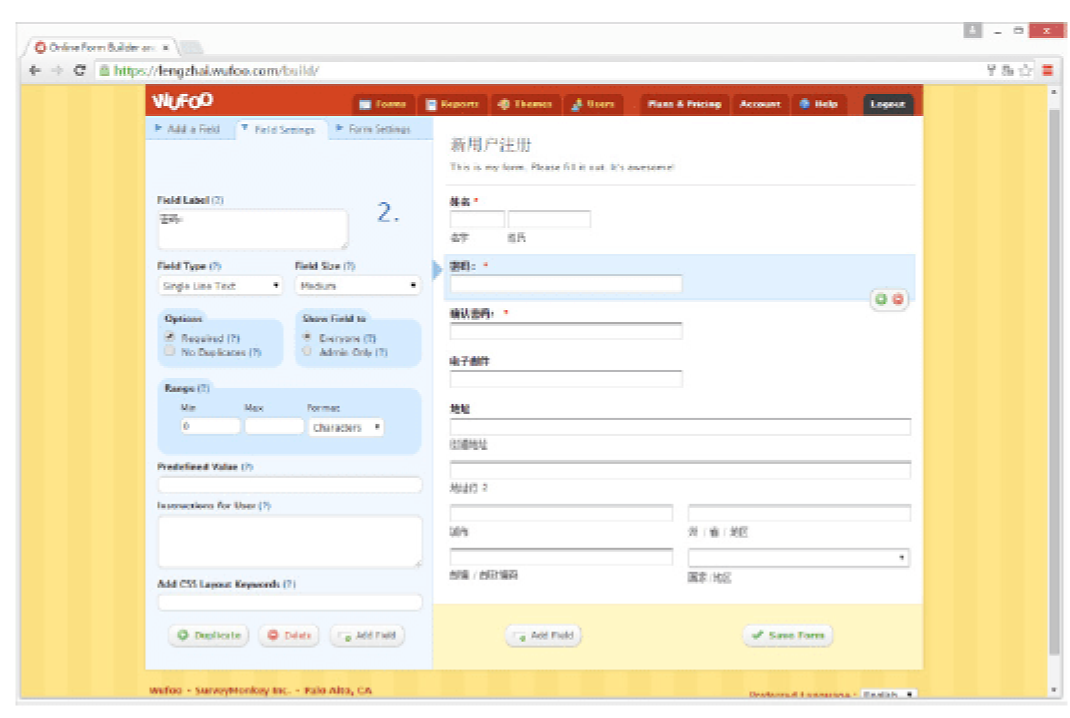


图9.14



9.5 制作表单页面

本例将制作一个用户注册的表单页面，帮助大家巩固和加强表单制作的技能。用户注册页面最终效果如图9.15所示。详细制作步骤如下：



图9.15

01 新建一个HTML页面，分析如图9.15所示的效果，该页面布局可分为上中下3部分结构，上面一个区域用于显示右上角的菜单，中间一个区域用于显示内容区域的菜单，下面一个区域用于显示表单、页面底部导航和版权等信息。根据以上分析，编写HTML代码如下所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>无标题文档</title>
</head>
<body>
<div id="container">
    <div id="header">
        <ul>
            <li><a href="#">投稿</a></li>
            <li><a href="#">登陆</a></li>
            <li><a href="#">注册</a></li>
        </ul>
    </div>
    <div id="ContentMenu">
        <ul>
            <li><a href="#">首页</a></li>
            <li><a href="#">看点</a></li>
            <li><a href="#">互联网</a></li>
            <li><a href="#">创投界</a></li>
            <li><a href="#">电商界</a></li>
            <li><a href="#">移动库</a></li>
            <li><a href="#">图说</a></li>
        </ul>
    </div>
    <div id="mainContent">
        <div id="regForm">
            <form>
                <ul>
                    <li><h3>注册帐户</h3></li>
                        <li><span class="labelSpan">登陆账号：</span><input
type="text" /><span class="noteSpan">*(可以使用中文，但禁止除[@][.]以外的特殊符号)</
span></li>
                         <li><span class="labelSpan">昵称：</span><input
type="text" /></li>
                        <li><span class="labelSpan">登陆密码：</span><input
type="password" /><span class="noteSpan">*</span></li>
                        <li><span class="labelSpan">确认密码：</span><input
type="password" /><span class="noteSpan">*</span></li>
                        <li><span class="labelSpan">电子邮箱：</span><input
type="email" /><span class="noteSpan">*(每个电子邮箱只能注册一个账号)</span></li>
                         <li><span class="labelSpan">验证码：</span><input
type="text" /><img src="img/img01.png"/></li>
                         <li><span class="labelSpan"> </span><input
```

```
type="submit" value="完成注册" /></li>
                    </ul>
                </form>
            </div>
            <div id="footerMenu">
                <ul>
                    <li><a href="#">关于我们</a> | </li>
                    <li><a href="#">加入我们</a> | </li>
                    <li><a href="#">广告及服务</a> | </li>
                    <li><a href="#">常见问题解答</a> | </li>
                    <li><a href="#">提交建议</a></li>
                </ul>
            </div>
            <div id="footer">
            <p>Copyright &copy; 2014-2015 D&C 版权所有</p>
            </div>
        </div>
    </div>
    </body>
    </html>
```

这段代码在浏览器中的效果如图9.16所示。



图9.16

02 这个页面中有3处使用了导航菜单，我们前面已经介绍过导航菜单的制作方法，这里就不再赘述，直接给出样式代码如下：

```
<style>
*{
margin:0;
padding:0;
}
#container{
```

```
margin:0 auto;
width:800px;
background-color:#F0F0F0;
padding-bottom:20px;
}
#header{
height:30px;
background-color:#2D2D2D;
font-size:12px;
color:white;
text-align:right;
padding-right:15px;
}
#header ul{
list-style:none;
display:inline-block;
}
#header ul li{
float:right;
line-height:30px;
font-family:'微软雅黑';
}
#header ul li a{
text-decoration:none;
font-size:12px;
color:white;
padding:3px 10px;
}
#header ul li a:hover{
background-color:#2291DE;
}
#mainContent{
height:400px;
width:650px;
margin:0 auto;
border:2px solid #E1E1E3;
background-color:white;
}
#ContentMenu{
text-align:center;
}
#ContentMenu ul{
list-style:none;
display:inline-block;
}
#ContentMenu ul li{
float:left;
line-height:30px;
font-family:'微软雅黑';
}
#ContentMenu ul li a{
```

```
text-decoration:none;
font-size:12px;
font-weight:bold;
color:#2D2D2D;
padding:3px 10px;
}
#footerMenu{
height:25px;
background-color:#2D2D2D;
font-size:12px;
color:white;
text-align:center;
}
#footerMenu ul{
list-style:none;
display:inline-block;
}
#footerMenu ul li{
float:left;
line-height:25px;
font-family:'微软雅黑';
}
#footerMenu ul li a{
text-decoration:none;
font-size:12px;
font-weight:bold;
color:white;
}
</style>
```

刷新页面后的效果如图9.17所示。



图9.17

03 下面设置form表单样式。首先要让form表单与周围其他元素分离开来，通过设置form表单的内边距即可达到这样的效果，相关代码如下：

```
#regForm form{
padding:30px 40px;
}
```

刷新页面后的效果如图9.18所示。



图9.18

04 表单中的各个元素使用列表显示，这里需要取消列表项前面的小黑点，然后设置列表项的字体大小、颜色和每个列表项距离下一个列表项的距离，相关代码如下：

```
#regForm form ul{
list-style:none;
}
#regForm form ul li{
font-size:12px;
color:#555555;
margin-bottom:13px;
}
```

刷新浏览器后的效果如图9.19所示。

图9.19

05 为了让表单中的标签对齐显示，我们设置每个span标签的宽度为60像素，并设置display属性为inline-block，让span元素在行内以块级元素显示，并设置文本向右对齐，相关代码如下所示：

```
#regForm ul li .labelSpan{
width:60px;
display:inline-block;
text-align:right;
}
```

刷新浏览器后的效果如图9.20所示。



图9.20

06 下面设置表单标题与提示信息的样式，以及图形验证码的尺寸，相关代码如下所示：

```
#regForm form ul li h3{
color:#2D2D2D;
}
#regForm ul li .noteSpan{
display:inline-block;
font-size:10px;
font-style:oblique;
}
#regForm ul li img{
width:60px;
height:20px;
vertical-align:bottom;
}
```

刷新浏览器后的效果如图9.21所示。



图9.21

07 form表单中默认的提交按钮样式与目前页面的风格很不相称，我们可以通过CSS样式对其进行修改，相关代码如下所示：

```
#regForm ul li input[type="submit"]{
background-color:#58AD44;
color:white;
font-size:13px;
font-weight:bold;
padding:8px 15px;
border-radius:4px;
border:hidden;
}
```

刷新浏览器后的效果如图9.22所示。

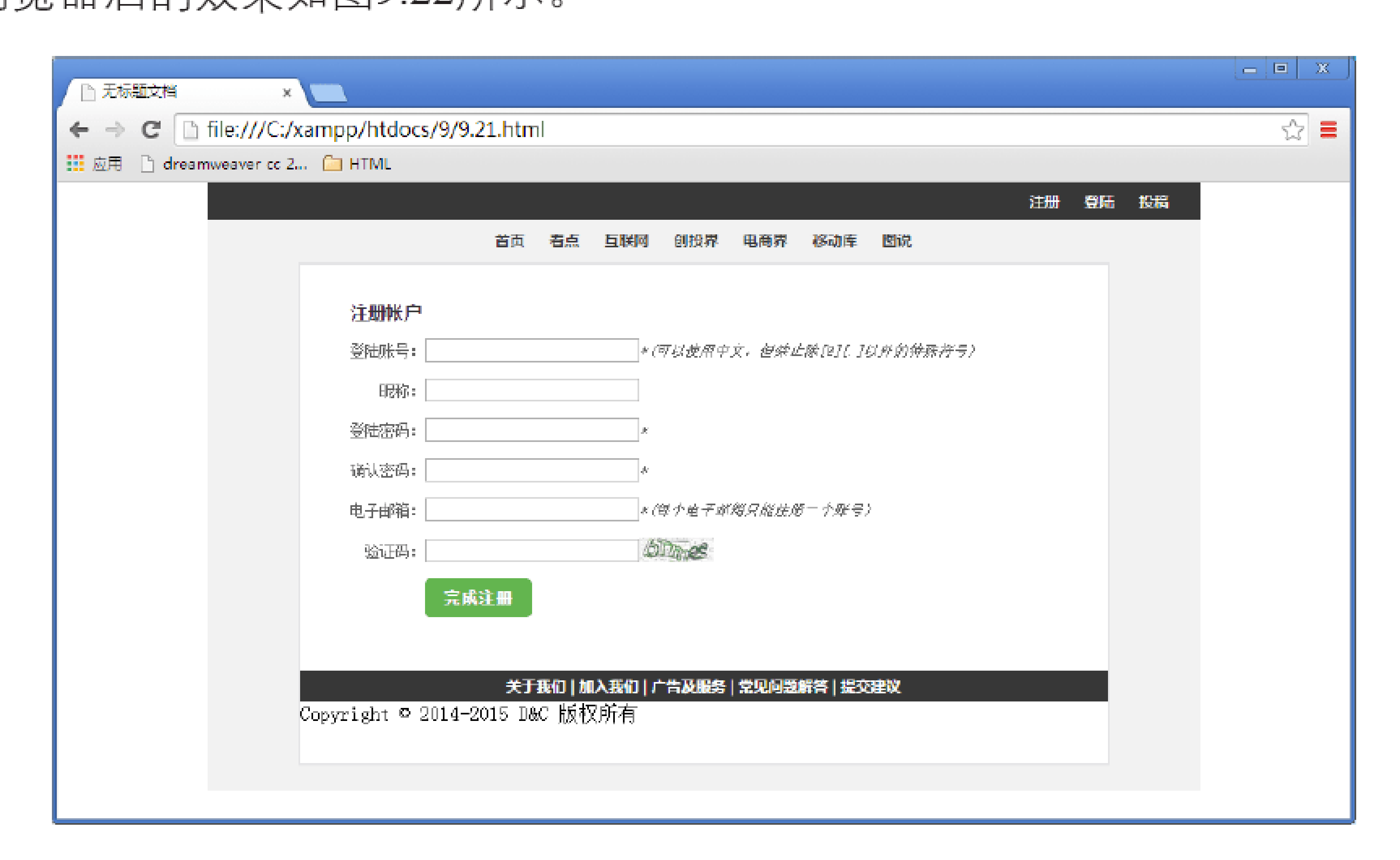


图9.22

08 最后通过CSS样式设置页面底部版权信息的样式，相关代码如下所示。

```
#footer{
height:50px;
text-align:center;
font-size:10px;
font-family:'微软雅黑';
}
#footer p{
display: inline-block;
line-height:50px;
}
```

刷新页面后的效果如图9.23所示。



图9.23

第10章 JavaScript基础

JavaScript是HTML页面中另一个非常重要的内容，早期很多的网页效果都是通过JavaScript来完成的.虽然目前HTML页面很多效果都可以通过CSS来完成，但是复杂一些的网页效果仍然需要使用JavaScript来完成，本章我们先来看一下JavaScript的基础知识。



10.1 JavaScript概述

JavaScript是目前Web应用程序开发者使用最广泛的客户端脚本编程语言，它不仅可用来开发交互式的Web页面，还可以将HTML、XML和Java Applet、Flash等功能强大的Web对象有机结合起来，使开发人员能快捷地生成Internet或Intranet上使用的分布式应用程序。另外，由于Windows给予其最为完善的支持并提供二次开发的接口来访问操作系统各组件并实施相应的管理功能，JavaScript成为继.bat（批处理文件）以来Windows系统里使用最为广泛的脚本语言。

JavaScript非常强大，下面我们来看一下JavaScript的特点。

（1）JavaScript是一种脚本语言，同样也是一种解释性语言，采用小程序段的方式实现编程，为用户提供了一个简单的开发过程。

（2）JavaScript是一种基于对象的语言，它可以创建和使用对象，许多功能都以对象为基础，通过操作对象的属性来实现相应的功能。

（3）简单性：首先它是一种基于Java基本语句和控制流之上的简单而紧凑的设计，对于学习Java是一种非常好的过渡。其次它的变量类型采用弱类型，并未使用严格的数据类型。

（4）安全性：它不允许访问本地的硬盘，并不能将数据存入到服务器上，不允许对网络文档进行修改和删除，只能通过浏览器实现信息浏览或动态交互，从而有效地防止数据的丢失。

（5）动态性：JavaScript可以直接对用户或者客户输入做出响应，无须经过Web服务程序。它对用户的反映响应，是采用以事件驱动的方式进行的。也就是当我们按下鼠标，移动窗口、选择菜单等事件发生时响应的。

（6）跨平台性：JavaScript依赖浏览器本身，与操作环境无关，只要能运行计算机，并安装有支持javascript的浏览器就可以正确执行。

10.2 JavaScript基本语法

每一种语言都有自己的语法，JavaScript也不例外。只有掌握了最基本的语法，才能按照自己的设计编写正确、高效的代码。

10.2.1 JavaScript书写方式

JavaScript有两种书写方式，一种是直接把JavaScript写在HTML页面里；另一种是新建一个JavaScript文件，将JavaScript代码写在JavaScript文件里，然后在HTML页面中通过script标签引入JavaScript文件。

直接写JavaScript的时候，可以将JavaScript写在<head>标签中，也可以写在<body>标签中，无论直接写在哪里，都需要将JavaScript代码写在<script>标签中。例如下面这段代码，<head>标签中的JavaScript代码用于输出一个字符串，而<body>标签中的JavaScript代码用于弹出一个提示信息，这段代码运行后的效果如图10.1所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>10.1</title>
<script>
document.writeln("欢迎来到JavaScript的世界！");
</script>
</head>
<body>
<script>
alert("这是网页中的内容！");
</script>
</body>
</html>
```

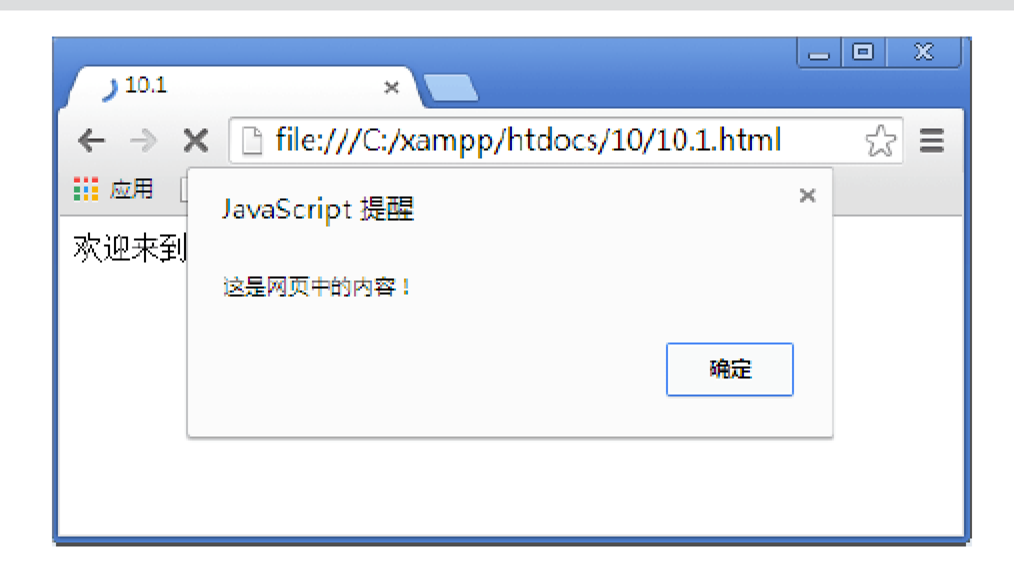


图10.1

JavaScript文件的后缀名是.js，如果将JavaScript代码写在JavaScript文件中，需要使用<script>标签的src属性指定这个JavaScript文件的路径，例如下面这段代码：

```
<script type="text/javascript" src="js/index.js"></script>
```

10.2.2 执行顺序与生命周期

JavaScript遵循自上而下的执行顺序，也就是说浏览器在解析JavaScript语言的时候，是按照JavaScript的书写顺序进行解析，如果将JavaScript的方法声明写在JavaScript调用之后，就会出现错误。通常情况下，如果直接将JavaScript写在HTML页面中，所有的JavaScript代码应该都写在<head>标签中。如有特殊需要，也可以将JavaScript写在<body>标签中，比如引用的JavaScript文件比较大，为了提高HTML页面的加载速度，可以将JavaScript引用写在<body>标签的最下面。

JavaScript定义的变量和函数只在当前页面有效，如果离开当前页面，这些变量和函数将无法访问，如果确实需要继续访问，可以考虑使用cookie。

10.2.3 变量

JavaScript中声明的变量都是弱类型变量，在声明变量的时候可以不考虑变量的类型，直接以var开头声明即可。在具体使用的时候，可以根据变量赋值的情况判断变量的类型。

如果将变量在函数体内部声明，就称为局部变量；如果在函数体外部声明，就称为全局变量。声明变量的时候，在同一个HTML页面中，尽量使用不同的变量名。在声明变量的同时，还可以给变量赋值。如果变量的值是字符串，则需要使用引号将值扩起来，JavaScript中不区分单引号和双引号。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>10.2</title>
<script>
var number;
var count=10;
var price=10.45;
var fromModel=’orm.Sean’;
var recList=new Object();
</script>
</head>
<body>
</body>
```

在这段代码中，变量number只声明了却没有赋值；变量count声明的同时赋值了一个整数；变量price声明的同时赋值了一个浮点数；变量fromModel声明的同时赋值了一个字符串，并且字符串用单引号括起来；变量recList声明的同时赋值了一个Object对象。

10.2.4 函数

JavaScript中用function声明函数，函数体必须写在大括号里面，JavaScript声明函数的语

法如下：

```
function 函数名(var1, var2... ){
    代码块...
}
```

其中函数名由用户自定义。但是要注意的是，因为在JavaScript中区分大小写，所以无论是声明函数还是调用函数，函数名的大小写都必须一致。小括号中的var1和var2表示函数的参数，如果函数不需要参数，也可以不写参数，但是小括号不能省略。

例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>10.2</title>
<script language="javascript">
function replaceUrl(strUrl){
document.getElementById("location").href=strUrl;
}
</script>
</head>
<body>
<a href="#" id="location" onMouseOver="replaceUrl('http://www.baidu.
com')">改变超链接地址</a>
</body>
</html>
```

在这段代码中声明了一个名为replaceUrl的函数，该函数有一个名为strURL的参数，通过<a>元素的id获取超链接对象，并将参数赋值给超链接对象的href属性。同时为超链接设置当鼠标移动到超链接上时执行该函数。运行这段代码，将鼠标移动到超链接上时，在浏览器窗口的左下角会显示出超链接的链接地址，效果如图10.2所示。

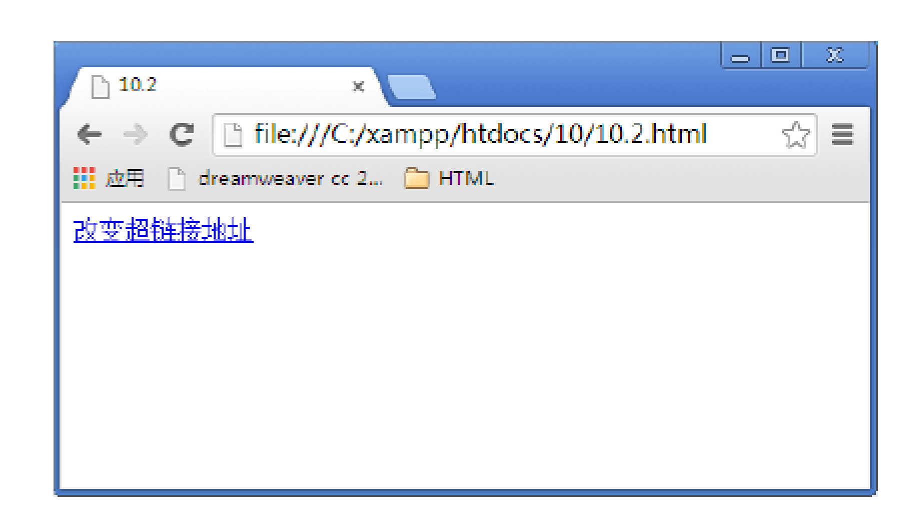


图10.2

该案例中调用JavaScript函数的方式，是在HTML元素的事件回调中执行函数，如果要在HTML页面加载的时候执行该函数，可以使用以下代码：

```
<body onload=" replaceUrl ('http://www.baidu.com')" >
```

另外还可以直接在超链接中执行调用函数的命令，相关代码如下：

```
<a href="javascript: replaceUrl ('www.baidu.com')" > 跳转页面</a>
```

最后还可以直接在JavaScript中执行函数调用，这种方法将在浏览器解析JavaScript代码的时候执行函数调用，相关代码如下：

```
<script>
       replaceUrl ("www.baidu.com");
</script>
```

10.2.5 类

需要明确的一点是，在JavaScript语法中并不支持类，但是为了完成某些复杂代码的编写，程序员们经过不懈的努力，研究出了如何用JavaScript模拟类。在JavaScript中，可以用function关键字来模拟创建一个类，例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>10.3</title>
<script>
function MenuItem(l, h, t)
{
    this.label = l;
    this.href = h;
    this.target = t;
    this.toHtml = function()
    {
        var html = "<a href='" + this.href + "' ";
        if(this.target != null){
            html += (" target='" + this.target + "' ");
        }
        html += " >" + this.label + "</a>";
        return html;
    }
}
</script>
</head>
<body>
</body>
</html>
```

在这段代码中使用JavaScript定义了一个名为MenuItem的类，类中使用this关键字表示实例的对象，label、href、target和toHtml都表示类中的属性，其中属性toHtml又是另一个函数的返回值，在这个函数中使用return返回了一个值。这样定义一个JavaScript类后，在使用的时候可以用以下代码代用这个类：

```
var item = new MenuItem("百度", "http://www.baidu.com", null);
```

10.2.6 Object类

在JavaScript中，我们还可以使用Object关键字来创建一个没有任何结构的类，在使用这个类的时候再为其添加成员变量，例如下面这段代码：

```
function createObject(){
    var obj=new Object();
    obj.name="objTest";
    obj.number=10;
    return obj;
}
```

在这段代码中，我们首先使用了new关键字创建了一个Object对象，然后为新创建的对象添加了name和number两个变量，并为其赋值，最后返回了这个对象。这就是使用Object来创建类的方法。

10.2.7 数组

在JavaScript中还可以使用Array关键字来创建一个数组，然后通过push方法为数组添加数据，例如下面这段代码：

```
<script>
var array=new Array();
for(var i=0;i<6;i++){
    array.push(i);
}
</script>
```

在这段代码中，我们使用Array关键字创建了一个名为array的数组，然后通过for循环，向这个数组中依次添加了6个数据。在使用数组的时候，我们可以使用数组变量名称加方括号，括号中带索引的方式来使用数组中的数据，相关代码如下：

```
var msg="";
for(var i=0;i<array.length;i++){
msg+="item:"+array[i]+'<br>';
}
document.writeln(msg)
```

在这段代码中，我们先定义了一个字符串变量，然后通过for循环，依次将array数组中的数据添加到msg变量中，并且对每个变量的数据进行二次处理，最后将msg数据展示在页面中。如图10.3所示为输出数组的效果。

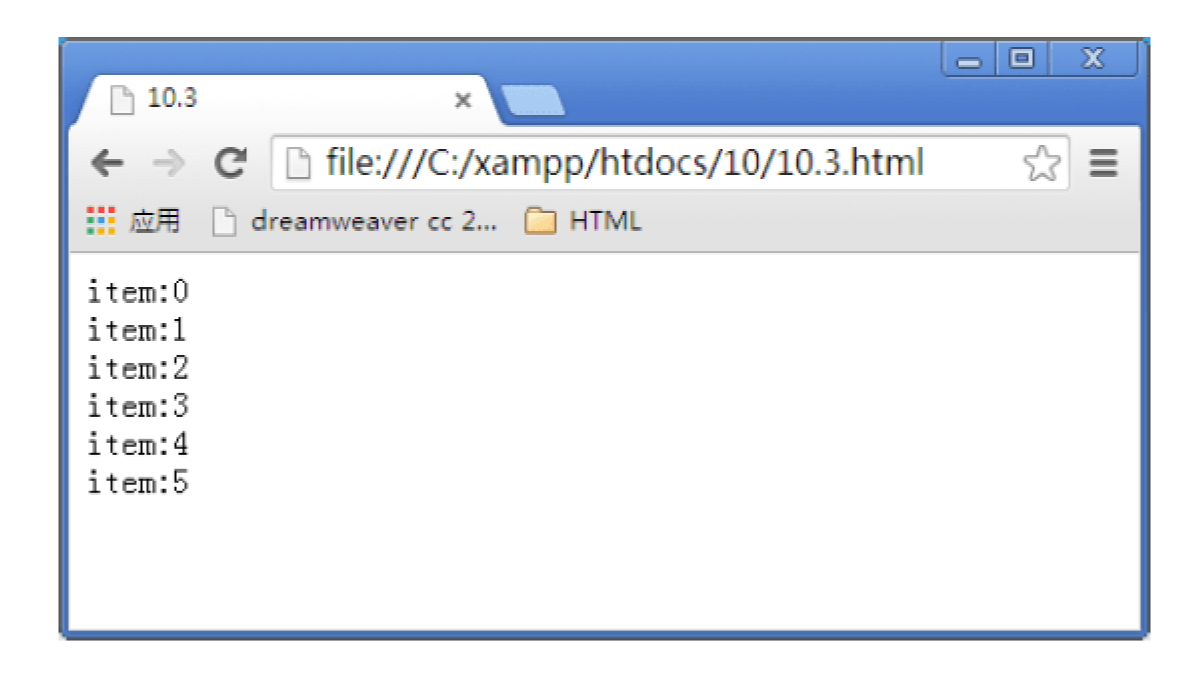


图10.3



10.3 使用JavaScript事件

定义了JavaScript函数，如果没有JavaScript事件进行驱动，这些函数仍然不能执行。那么什么是JavaScript事件呢？当我们刷新Web页面的时候，Web页面会触发onLoad事件；当我们点击页面上的按钮的时候会触发onClick事件；当我们把鼠标移动到某个控件上面的时候，会触发onMouseOver事件。在Web页面上所有的这些动作，都会触发相应的事件。JavaScript中常用的事件如下表所示：

表10.1

属性	描述
onClick	鼠标单击事件
onMouseOver	鼠标经过事件
onMouseOut	鼠标移出事件
onChange	文本框内容改变事件
onSelect	文本框内容被选中事件
onFocus	获取焦点事件
onBlur	失去焦点事件
onLoad	页面加载事件
onUnload	关闭页面事件

例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>10.4</title>
<script language="javascript">
function init(){
document.getElementById("msg").innerHTML="请在文本框中输入信息。";
}
function mouseOver(){
document.getElementById("msg").innerHTML="点击按钮后，文本框中的信息将显示在这里。";
}
function textChange(){
document.getElementById("msg").innerHTML=document.getElementById("txt").value;
}
function onclick(){
document.getElementById("msg").innerHTML=document.getElementById("txt").value;
```

```
}
</script>
</head>
<body onLoad="init()">
<h2>JavaScript测试</h2>
请输入:
<input type="text" id="txt" onChange="textChange()"
onMouseMove="mouseOver()" />
<input id="btn" type="button" value="按钮" onClick="onclick()"/>
<br>
<h2 id="msg"></h2>
</body>
</html>
```

在这段代码中，总共有4个函数，函数init用于当页面加载的时候显示提示信息；函数mouseOver用于当鼠标经过文本框时显示提示信息；函数textChange用于当文本框中的内容发生改变时，将改变后的内容显示在提示信息中；函数onclick用于当点击按钮的时候，将文本框中的信息显示在提示信息中。这4个函数分别在<body>元素的onLoad事件、<input>元素的onChange事件、onMouseMove和onClick事件中调用。页面加载时的效果如图10.4所示。当鼠标移动到文本框时的效果如图10.5所示。在文本框中输入文字，点击按钮后的效果如图10.6所示。

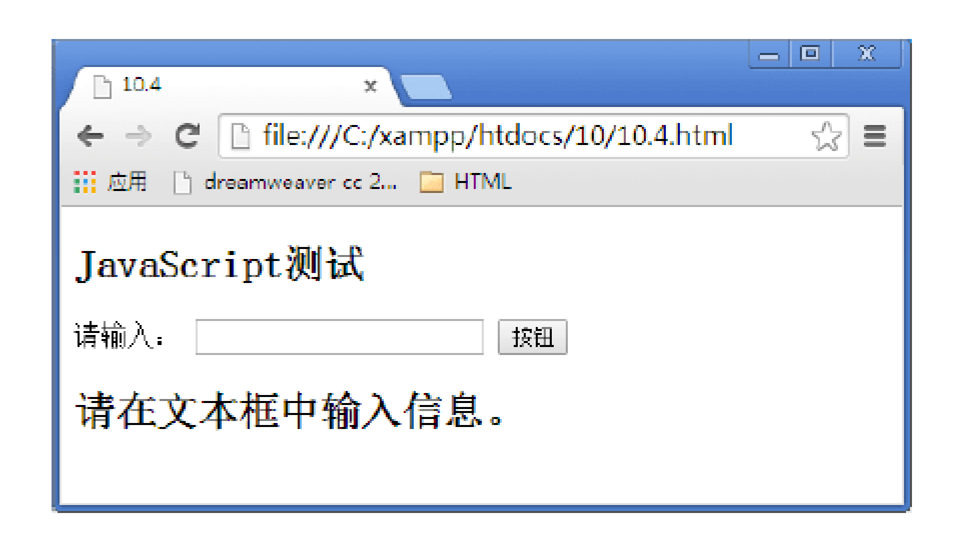


图10.4



图10.5

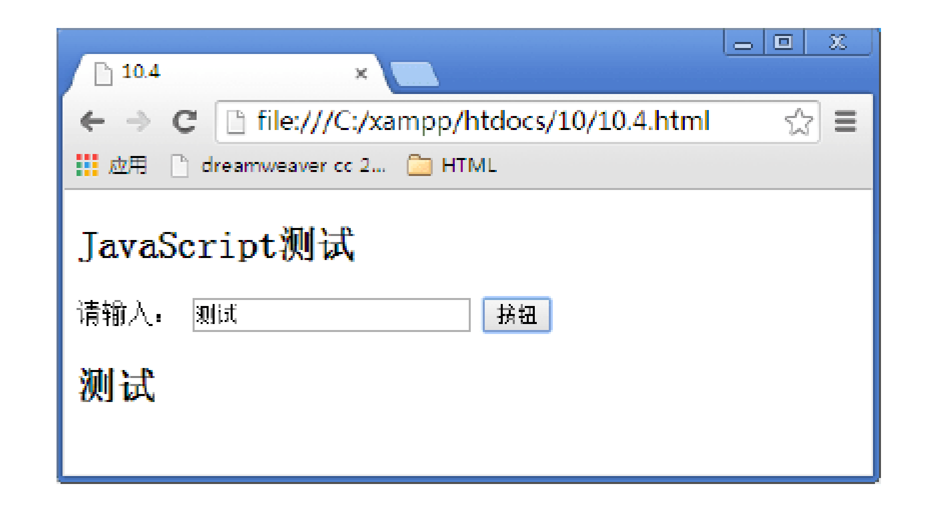


图10.6

第11章 使用JavaScript创建交互式网页

我们通常将浏览器称为Web前端，将Web服务器称为后端，用户在前端的操作，在后端会有对应的程序进行处理。这样，一个过程就形成了前后端的一次交互。然而在实际应用中，有很多情况并不需要后端做出判断，直接在前端就可以完成判断，比如用户输入的电子邮件地址是否有效等。下面我们介绍浏览器前端交互的相关内容。



11.1 常用JavaScript特效

JavaScript虽然是一些代码段，但是经过这么多年的发展和积累，已经具有了很多独特的功能，使用这些功能就可以完成一些特殊的效果。

11.1.1 时间日期特效

在JavaScript中使用Date关键字创建日期对象，通过日期对象就可以获取当前系统时间，以及时针、分针、秒针、星期几、日期、月份和年份。有了这些对象，我们就可以创建各种时间日期特效。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>11.1</title>
</head>
<body>
<script language="javascript">
function getDayCN(day){
var value="";
switch(day){
case 1:
value="星期一";
break;
case 2:
value="星期二";
break;
```

```
case 3:
value="星期三";
break;
case 4:
value="星期四";
break;
case 5:
value="星期五";
break;
case 6:
value="星期六";
break;
case 7:
value="星期天";
break;
default:
value="";
break;
}
return value;
}
var dateObj=new Date();
var msg="<h2>当前时间</h2>";
msg+="<b>";
msg+=dateObj.getUTCFullYear()+"年";
msg+=dateObj.getMonth()+1+"月";
msg+=dateObj.getDate()+"日";
msg+=dateObj.getHours()+"时";
msg+=dateObj.getMinutes()+"分";
msg+=dateObj.getSeconds()+"秒";
msg+=dateObj.getMilliseconds()+"毫秒";
msg+="  "+getDayCN(dateObj.getDay());
msg+="</b>"
document.writeln(msg);
</script>
</body>
</html>
```

在这段代码中，JavaScript函数getDayCN通过传进来的参数返回星期，使用Date关键字创建了一个dateObject对象，然后根据这个对象分别获取年、月、日、时、分、秒、毫秒和星期。需要注意的是，月份的索引从0开始，因此返回正确的月份需要再加1。对于年来说，标准浏览器都是以1900年为起始年，再加上dateObj.getYear()获取的年份，就是当前的年份；而在IE中，dateObj.getYear()将直接获取当前的年份。

在下面这段代码中，通过JavaScript中的时间函数创建了一个动态的时间效果。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>11.2</title>
```

```
</head>
<style type="text/css">
#showtime {
margin:0 auto;
background:#5C1A1C;
height: 30px;
vertical-align:middle;
line-height: 30px;
width: 220px;
border: 2px #CB4F51 solid;
}
#localtime {
color:white;
font-size:18px;
font-weight:bold;
}
</style>
<body>
<div id="showtime"> <span id="localtime"></span> </div>
<script language="javascript">
function showLocale(objD) {
var str;
var hh = objD.getHours();
var mm = objD.getMinutes();
var ss = objD.getSeconds();
var month = objD.getMonth()+1;
var date = objD.getDate();
if(hh<10)hh=’ 0’ +hh;
if(mm<10)mm=’ 0’ +mm;
if(ss<10)ss=’ 0’ +ss;
if(month<10)month=’ 0’ +month;
if(date<10)date = ‘0’ + date;
var year = objD.getFullYear();
str=year+"年";
str+=month+"月"+date+"日 ";
str+= hh+":"+mm+":"+ss;
return(str);
}
function tick() {
var today;
today=new Date();
document.getElementById("localtime").innerHTML=showLocale(today);
}
window.setInterval("tick()",1000);
</script>
</body>
</html>
```

运行这段代码后，效果如图11.1所示。这个时间与当前系统时间保持一致，并且每秒钟更新一次。

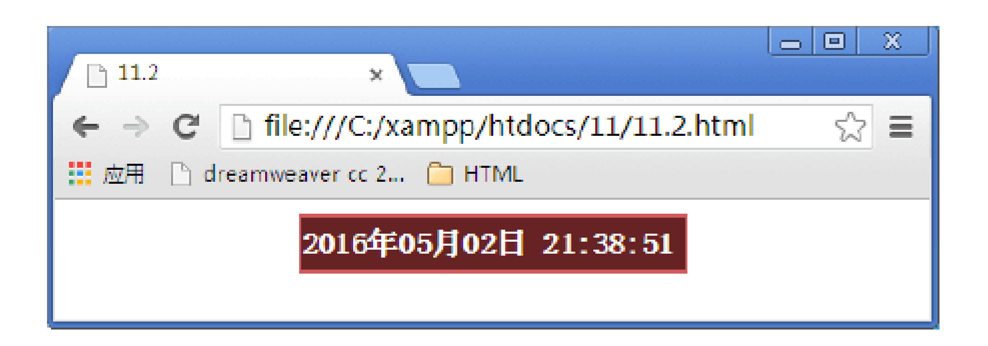


图11.1

11.1.2 页面特效

JavaScript还可以用来控制很多页面特效，比如进度条效果、导航菜单效果、随机显示导航效果等。下面我们就来看一个进度条的效果，相关代码如下：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>11.3</title>
<style>
#loading{
color:#0066ff;
size:2;
face="Arial";
}
#chart{
font-family:Arial;
font-weight:bold;
color:#0066ff;
background-color:#fef4d9;
padding:0px;
border-style:none;
}
#percent{
color:#0066ff;
text-align:center;
border-width:medium;
border-style:none;
}
</style>
</head>
<body>
<form name=loading>
  <p align=center id="loading">载入中，请稍等...<br>
    <input type=text id="chart" name=chart size=46 readonly ><br>
    <input type=text id="percent" name=percent size=47 readonly >
<script>
var bar=0
var line="||"
var amount="||"
```

```
count()
function count(){
bar=bar+2
amount =amount + line
document.loading.chart.value=amount
document.loading.percent.value=bar+"%"
if (bar<99)
{setTimeout("count()",100);}
else
{window.location = "#";}
}</script>
</p>
</form>
</body>
</html>
```

在这段代码中有1个form表单，表单中的<p>标签中内嵌了两个<input>标签，第1个<input>标签用于显示进度条状态，第2个<input>标签用于显示进度的百分比。函数count用于计算当前进度条的进度和状态，每100毫米计算一次，如果进度条没有达到100，则继续为第1个<input>标签中添加竖线，同时更改百分比，否则跳转到新的地址。运行这段代码后，页面效果如图11.2所示。

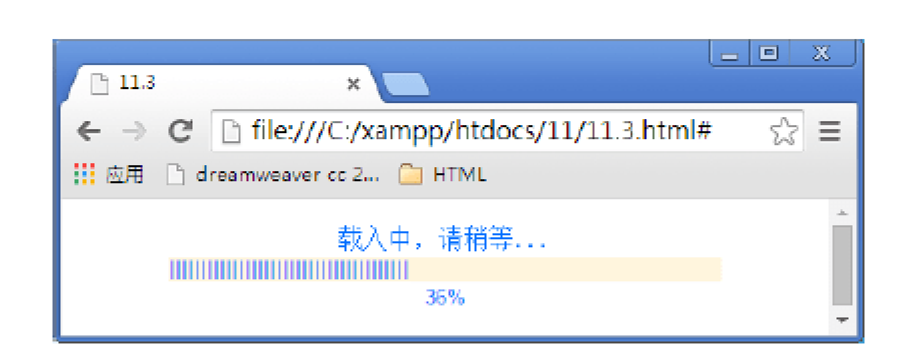


图11.2

11.1.3 图形图像特效

使用JavaScript还可以控制图像的一些特效，例如鼠标点击的效果、图像循环显示的效果、图像渐渐出现的效果、图像变形扭曲的效果等。下面我们就来看一个图形变形扭曲的效果，相关代码如下：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>11.4</title>
</head>
<body onLoad="fade()">
<img src="img/img01.jpg"  name="u" >
<script language="JavaScript">
var b = 1;
var c = true;
function fade(){
if(document.all);
```

```
if(c == true) {
b++;
}
if(b==100) {
b--;
c = false
}
if(b==10) {
b++;
c = true;
}
if(c == false) {
b--;
}
u.width=150 + b;
u.height=125 - b;
setTimeout("fade()",50);
}
</script>
</body>
</html>
```

在这段代码中，HTML页面中只有一个图像标签，用于显示一幅图像。而在JavaScript代码中，每50毫秒就会执行一次fade函数，这个函数的主要功能就是不断变换图像的长和宽。图像的初始尺寸设置长度为150、高度为125，通过一个变量b来控制图像的尺寸，如果b等于100，就开始减小b，并将其设置到图像的尺寸上，如果b等于10，就开始增大b，并将其设置到图像的尺寸上。运行这段代码后，效果如图11.3所示，图像的长和宽逐渐缩小，缩小到一定程度后又开始逐渐增大。

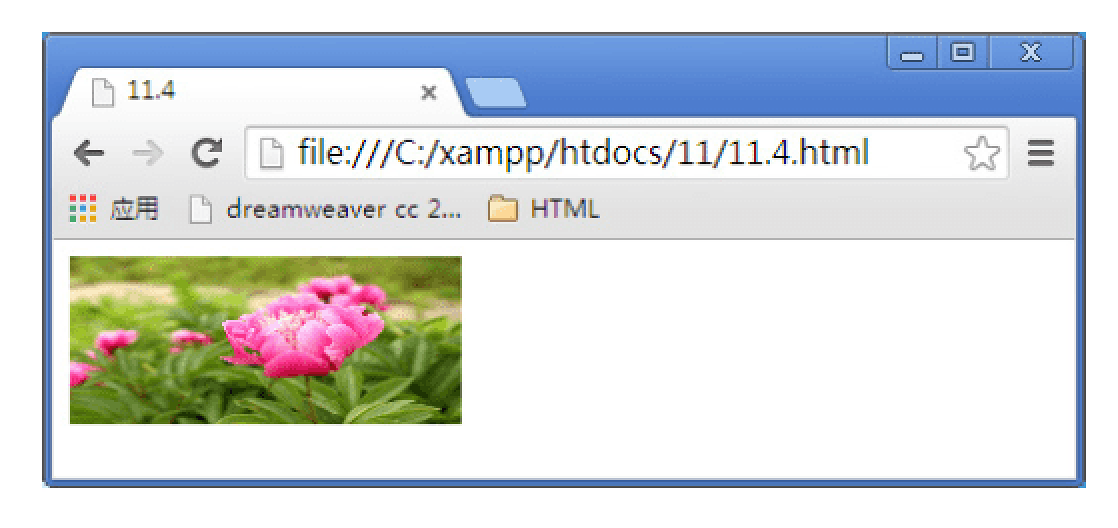


图11.3

11.1.4 页面导航特效

使用JavaScript控制页面导航特效的案例也有很多，比如多级滚动的菜单、可移动的显示层、无边框的广告窗口、可拖动的菜单等。下面我们来看一个通过下拉菜单打开页面链接的导航效果，相关代码如下所示：

```
<!doctype html>
<html>
<head>
```

```
    <meta charset="utf-8">
    <title>11.5</title>
    </head>
    <body>
    <script language="javascript">
    function gothere(){
    var thebox=document.mycombowopt
    if (thebox.windowoption.checked){
    if (!window.newwindow)
    newwindow=window.open("")
    newwindow.location=thebox.example.options[thebox.example.selectedIndex].
value
    }
    else
    location=thebox.example.options[thebox.example.selectedIndex].value
    }
    </script>
    <form name="mycombowopt">
    <select name="example" size=1>
    <option value="http://www.163.com">网易</option>
    <option value="http://www.sohu.com">搜狐</option>
    <option value="http://www.sina.com.cn">新浪</option>
    <option value="http://www.baidu.com">百度</option>
    <option value="http://cn.bing.com/">Bing</option>
    </select> <input type="button" value="Go!" onClick="gothere()"> <br>
    <input type="checkbox" name="windowoption" value="ON">新窗访问</p>
    </form>
    </body>
    </html>
```

在这段代码中有一个form表单，表单中内嵌了一个下拉选择框、一个<input>按钮和复选框。当点击按钮的时候，执行gothere函数，该函数用于跳转到当前选中项对应的链接地址；如果同时勾选了复选框，则在新窗口中打开选中项对应的链接地址。运行这段代码后，选择一个下拉选项，点击按钮后即可导航到新的页面。效果如图11.4所示。

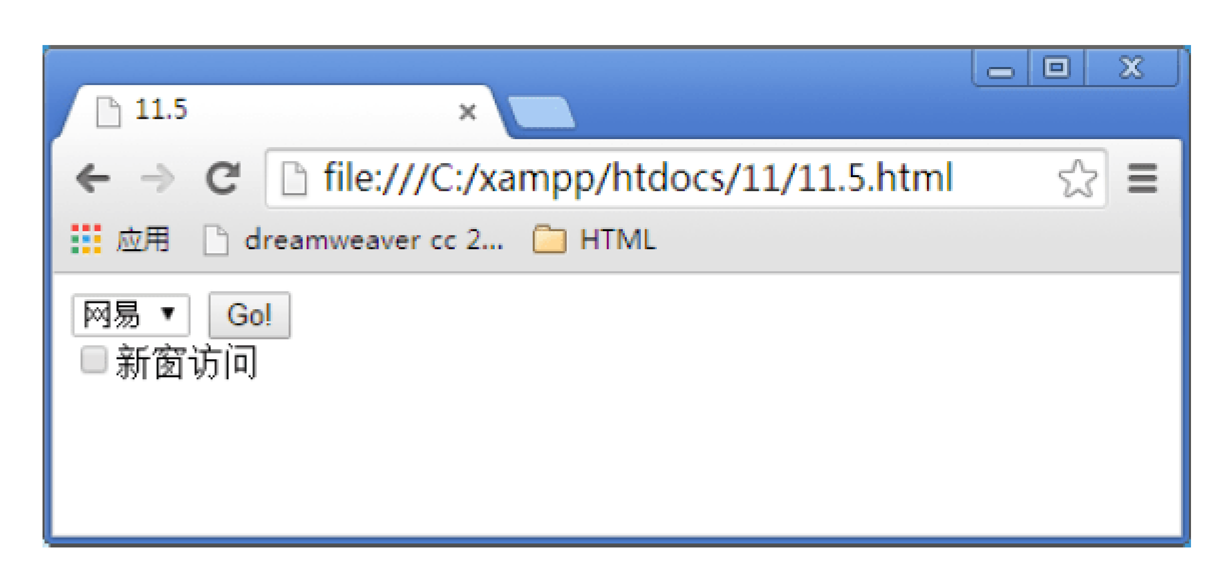


图11.4

11.1.5 文本特效

使用JavaScript控制文本的特效也有很多种，例如文本自动输出、文字扭动播放、多条

文字播放特效、字符从右波浪式出现然后下落等效果。下面我们来看一个随机显示信息的特效，相关代码如下：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>11.6</title>
<script language="JavaScript">
var a = Math.random() + ""
var rand1 = a.charAt(5)
quotes = new Array
quotes[1] = '这是一个很长的文本信息'
quotes[2] = '这也许是你见过的最无聊的信息了'
quotes[3] = '哈哈'
quotes[4] = '真是个不错的选择'
quotes[5] = '这是谁的注意'
quotes[6] = '明天早上来我办公室'
quotes[7] = '夕阳西下'
quotes[8] = '这是一个互联网的时代'
quotes[9] = '针灸'
quotes[0] = '明天早上的太阳'
var quote = quotes[rand1]
document.write( quote )
</script>
</head>
<body>
</body>
</html>
```

这段代码在JavaScript中创建了一个数组，并且给数组中添加了很多数据，然后通过一个random随机数获取到一个索引数，并将数组中位于这个索引的数据显示在页面上。运行这段代码后，重复刷新页面会得到不同的效果，如图11.5所示。

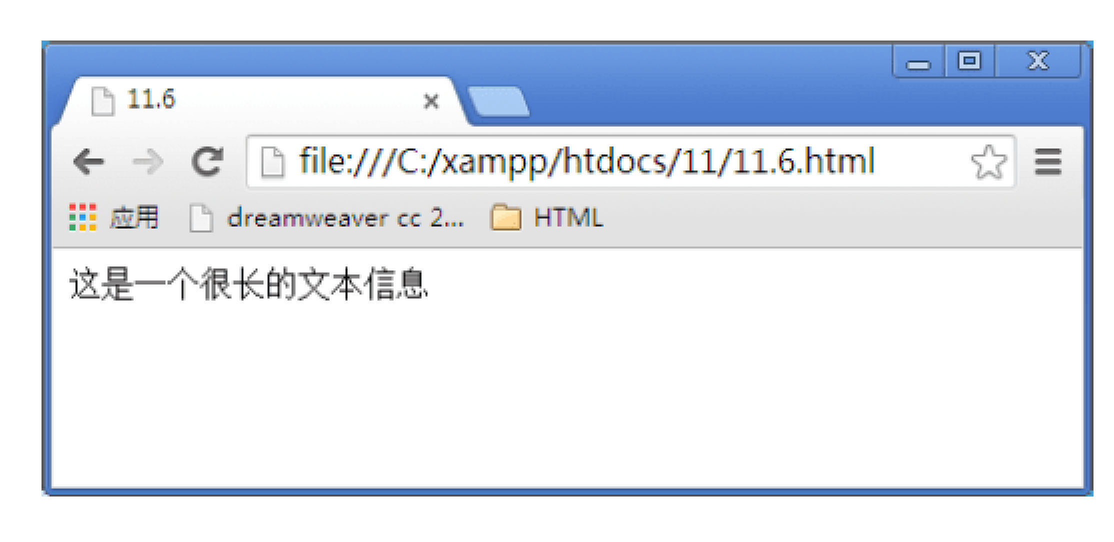


图11.5

11.1.6 鼠标特效

使用JavaScript控制鼠标特效的用法很多，比如跟随鼠标的花絮、各种形状的鼠标、禁止鼠标左键、鼠标点击效果、三色鼠标等。下面我们来看一个显示鼠标坐标的特效，相关代码

如下：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>11.7</title>
</head>
<body>
<div>
    <p id="x"></p>
    <p id="y"></p>
</div>
<script type="text/javascript">
var x , y;
function positionBody(event){
event = event||window.event;
x=event.clientX
y=event.clientY
}
document.onmousemove = function(event){
positionBody(event);
document.getElementById("x").innerHTML = "x= " + x + "px";
document.getElementById("y").innerHTML = "y= " + y + "px";
}
</script>
</body>
</html>
```

在这段代码中，页面只有两个<p>标签，分别用于记录当前鼠标的x和y坐标。当鼠标移动的时候执行函数，通过positionBody函数获取当前鼠标相对于body定位的x和y坐标，然后赋值给<p>标签，运行这段代码后，效果如图11.6所示。

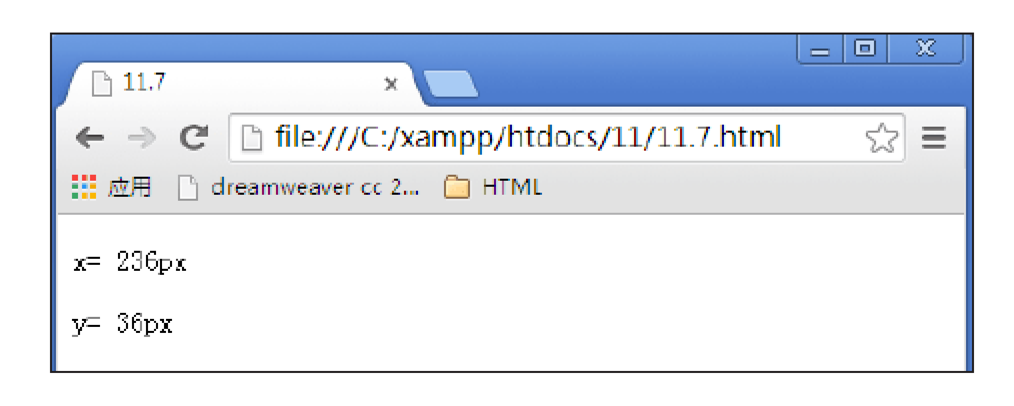


图11.6



11.2 防止访客刷新内容

当用户提交表单数据之后，如果再次刷新页面，有可能出现再次提交数据的情况。为了

避免这类事情的发生，在编写程序的时候需要做一些特殊处理。针对这类情况的处理方案很多，下面我们就来介绍一下如何使用JavaScript防止用户刷新页面。

11.2.1 禁用F5刷新

当页面触发onkeydown事件的时候，可以通过获取当前键盘按钮来控制禁用F5功能键，从而禁用F5刷新功能。相关代码如下：

```
<script language="javascript">
document.onkeydown=function(){
if ( event.keyCode==116){
event.keyCode = 0;
event.cancelBubble = true;
return false;
}
}
</script>
```

在这段代码中，使用event.keyCode获取按下按键的ACSII码，F5的ACSII码是116，如果键盘上按下的是F5键，那么设置event.keyCode=0，并返回一个false，这样就可以禁止F5的刷新功能了。将这段代码应用到页面中，按下F5功能键将没有任何反应。

11.2.2 禁止右键弹出菜单

在页面上单击鼠标右键，也可以看到一个刷新功能，为了防止用户从这里刷新页面，我们可以通过禁止右键弹出菜单的方法禁止刷新功能。相关代码如下：

```
<script language="javascript">
document.oncontextmenu=function(){
      return false;
}
</script>
```

将这段代码添加到页面中，当在页面上单击鼠标右键的时候，右键菜单将不会弹出。

11.2.3 屏蔽其他功能

使用JavaScript还可以在页面中屏蔽很多其他的功能，这里我们介绍一些比较有用的使用JavaScript屏蔽的功能。

屏蔽F1功能键的帮助功能可以使用下面的代码：

```
window.onhelp=function(){
 return false
}
```

屏蔽Alt+方向键的功能可以使用下面的代码：

```
document.onkeydown=function(){
```

```
    if ((window.event.altKey) && ((window.event.keyCode==37)|| (window.event.keyCode==39))) {
    alert("禁用ALT+方向键前进或后退网页！");
    event.returnValue=false;
    }
    }
```

ACSII为37表示向左的的方向键，ACSII为39表示向右的方向键。如果在页面上使用退格删除键也可以达到页面返回的效果，退格删除键的ACSII为8，使用下面的代码同样可以屏蔽退格删除键功能。

```
    document.onkeydown=function(){
    if (window.event.keyCode==8) {
    alert("禁用退格删除键！");
    event.returnValue=false;
    }
    }
```

F11功能键可以让浏览器全屏显示，其对应的ACSII码是122；Ctrl+n组合键可以新打开一个浏览器窗口，n的ACSII码为78；Alt+F4组合键可以关闭当前打开的窗口，F4功能键的ACSII码是115；其他键盘上的按钮对应的ACSII码可以查看ACSII对照表，使用以上介绍的JavaScript方法屏蔽相关功能。



11.3 使用jQuery

JavaScript对于改善网页效果起到了很大的帮助，但是我们在编写程序的时候，经常会因为编写很多重复的代码块，让整个文件变得臃肿，运行变得缓慢，而且还会出现很多错误。为了改善这种状况，可以使用jQuery帮助我们优化代码。

11.3.1 什么是jQuery

jQuery是一个兼容多浏览器的JavaScript函数库，使用jQuery编写的代码不但精简而且功能更强。jQuery函数库包含HTML元素的选取和操作、CSS操作、HTML事件函数、JavaScript特效和动画、HTML DOM遍历和修改、AJAX和Utilities，它是一个辅助JavaScript开发的非常有用的函数库。

例如，页面中有一个id为note的元素，如果使用JavaScript获取这个对象，我们应该使用下面这句代码：

```
    var obj=document.getElementById("note");
```

如果我们使用jQuery获取这个对象，就非常方便了，代码如下：

```
    var obj=$("p");
```

11.3.2 如何应用jQuery

jQuery是一个JavaScript函数库，在使用jQuery之前，需要在页面中引入JavaScript函数库，比如谷歌和微软的服务器上都有jQuery，它们的引入方法如下。

从谷歌引入jQuery的方法如下所示：

```
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.8.0/jquery.min.
js"></script>
```

从微软引入jQuery的方法如下所示：

```
    <script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.8.0.js"></
script>
```

另外，我们还可以直接从jQuery官网（http://jquery.com/download/）上下载最新的jQuery库，然后将jQuery库和我们的HTML文件放在一起，这样就可以直接在HTML页面中引用了。

引入jQuery之后，我们可以使用$符号，配合HTML元素选取页面中的特定元素，然后执行相关的操作，例如下面这段代码：

```
    <!doctype html>
    <html>
    <head>
    <meta charset="utf-8">
    <title>11.9</title>
    <script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.8.0.js"></
script>
    <script language="javascript">
    function hideTitle(){
    $("#title").hide();
    }
    function hideP(){
    $(".p").hide();
    }
    function hideSpan(){
    $("span").hide();
    }
    </script>
    </head>
    <body>
    <h2 id="title">标题</h2>
    <p class="p">这是段落</p>
    <span>这是内容</span><br>
    <input type="button" value="隐藏标题" onClick="hideTitle()"/>
    <input type="button" value="隐藏段落" onClick="hideP()"/>
    <input type="button" value="隐藏内容" onClick="hideSpan()"/>
    </body>
    </html>
```

在这段代码中，HTML页面中有<h2>、<p>和<span>这些元素，<h2>元素设置了id属性，<p>元素设置了class属性，<span> 元素没有设置任何属性。这些元素下面有3个按钮，分别用于隐藏对应的HTML元素。在hideTitle函数中使用id属性获取HTML元素，并将其隐藏；在hideP函数中使用class属性获取HTML元素，并将其隐藏；在hideSpan函数中直接使用HTML元素标记获取HTML元素，并将其隐藏。

使用jQuery之后，我们获取对象的方法仍然非常方便，而且jQuery的功能远远不止这么简单。由于篇幅的关系，这里仅介绍jQuery简单的使用方法，有关jQuery更多的操作，可以查询相关资料。



11.4 使用bootstrap

jQuery是一个非常经典的辅助JavaScript开发的函数库，但是一些不甘平庸的工程师为了提高他们内部的分析和管理能力，仍然用业余时间构建了一套易用、优雅、灵活和可扩展的前端工具集，这就是bootstrap，并且越来越多的工程师为其共享了高质量的代码和文档。

11.4.1 什么是bootstrap

bootstrap最早是由一些twitter工程师构建的，其目的在于提高他们内部的分析和管理能力，后来开源之后，越来越多的工程师加入了bootstrap大军，为其贡献高质量的代码和文档。同时还涌现了许多基于bootstrap建设的网站，这些网站不仅界面清新、简洁，而且排版利落大方，例如图11.7和图11.8所示的两个网站。

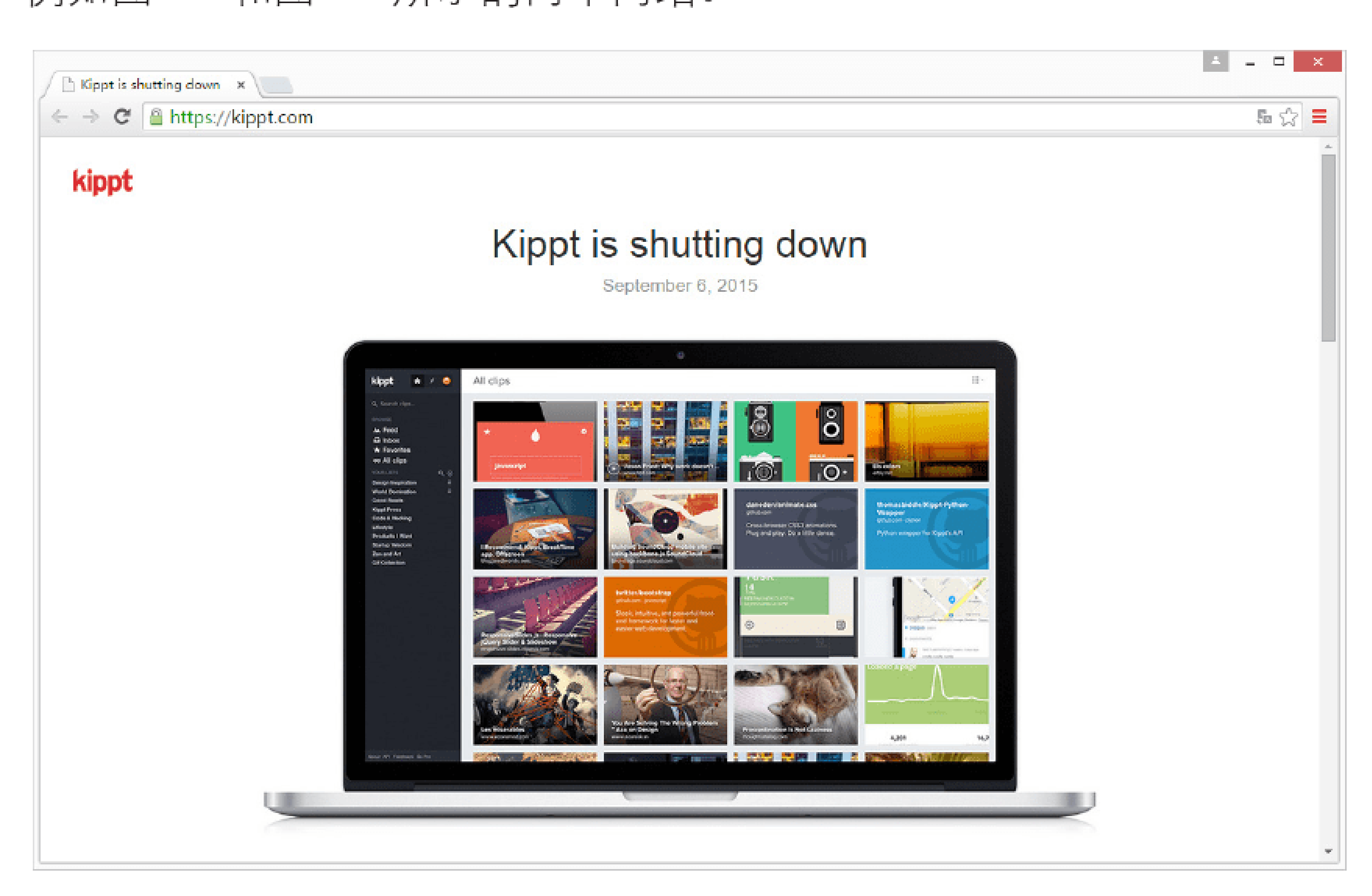


图11.7

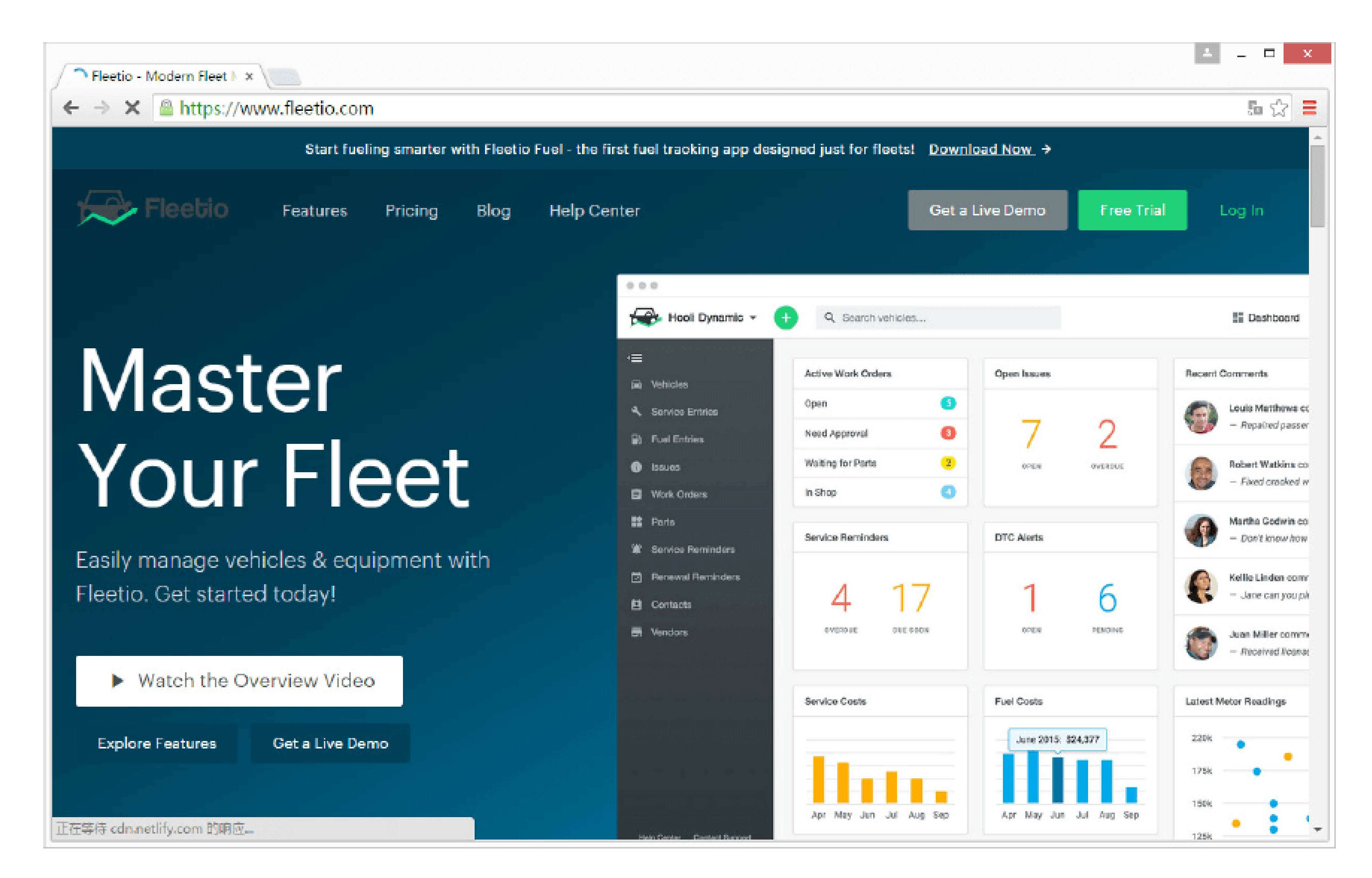


图11.8

目前所有的主流浏览器都支持bootstrap，它的框架包含了贯穿于整个库的移动设备优先的样式。只要具备HTML和CSS的基础知识，就可以开始学习bootstrap，而且bootstrap的响应式CSS能够自适应于台式机、平板电脑和手机。

基于开源项目的bootstrap，为开发人员提供了一个简洁统一的解决方案，而且它还包含了功能强大的内置组件，易于定制。

11.4.2 如何应用bootstrap

bootstrap的使用方法和jQuery类似，需要在HTML页面中引入bootstrap资源，推荐使用百度的静态资源和CDN服务，这样访问速度更快、加速效果更明显、没有速度和带宽限制。首先需要引入bootstrap核心的CSS文件，相关代码如下：

```
    <link href="http://apps.bdimg.com/libs/bootstrap/3.3.0/css/bootstrap.min.
css" rel="stylesheet">
```

引入可选的bootstrap主题文件，相关代码如下：

```
    <script src="http://apps.bdimg.com/libs/bootstrap/3.3.0/css/bootstrap-
theme.min.css"></script>
```

引入jQuery文件，注意必须在bootstrap.min.js之前引入，相关代码如下：

```
    <script src="http://apps.bdimg.com/libs/jquery/2.0.0/jquery.min.js"></
script>
```

最后引入bootstrap核心的JavaScript文件，相关代码如下：

```
    <script src="http://apps.bdimg.com/libs/bootstrap/3.3.0/js/bootstrap.min.
js"></script>
```

在HTML页面中，经常会看到一些带图标的按钮，为了实现这些效果，以往的做法是找

一幅图像作为按钮的背景，但是如果使用bootstrap，就不需要使用图像，可以直接使用字体图标，例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>11.10</title>
<link href="http://libs.baidu.com/bootstrap/3.0.3/css/bootstrap.min.css"
rel="stylesheet">
<script src="http://libs.baidu.com/jquery/2.0.0/jquery.min.js"></script>
<script src="http://libs.baidu.com/bootstrap/3.0.3/js/bootstrap.min.js"></
script>
</head>
<body>
<button type="button" class="btn btn-default btn-lg"> <span
class="glyphicon glyphicon-user"></span> User </button>
<button type="button" class="btn btn-default btn-sm"> <span
class="glyphicon glyphicon-user"></span> User </button>
<button type="button" class="btn btn-default btn-xs"> <span
class="glyphicon glyphicon-user"></span> User </button>
</body>
</html>
```

在这段代码中有3个按钮，我们并没有为其设置特定的背景图标，只是设置了按钮的class属性和<span>标签的class属性，这样我们就创建了3个尺寸不一样的按钮图标，如图11.9所示。

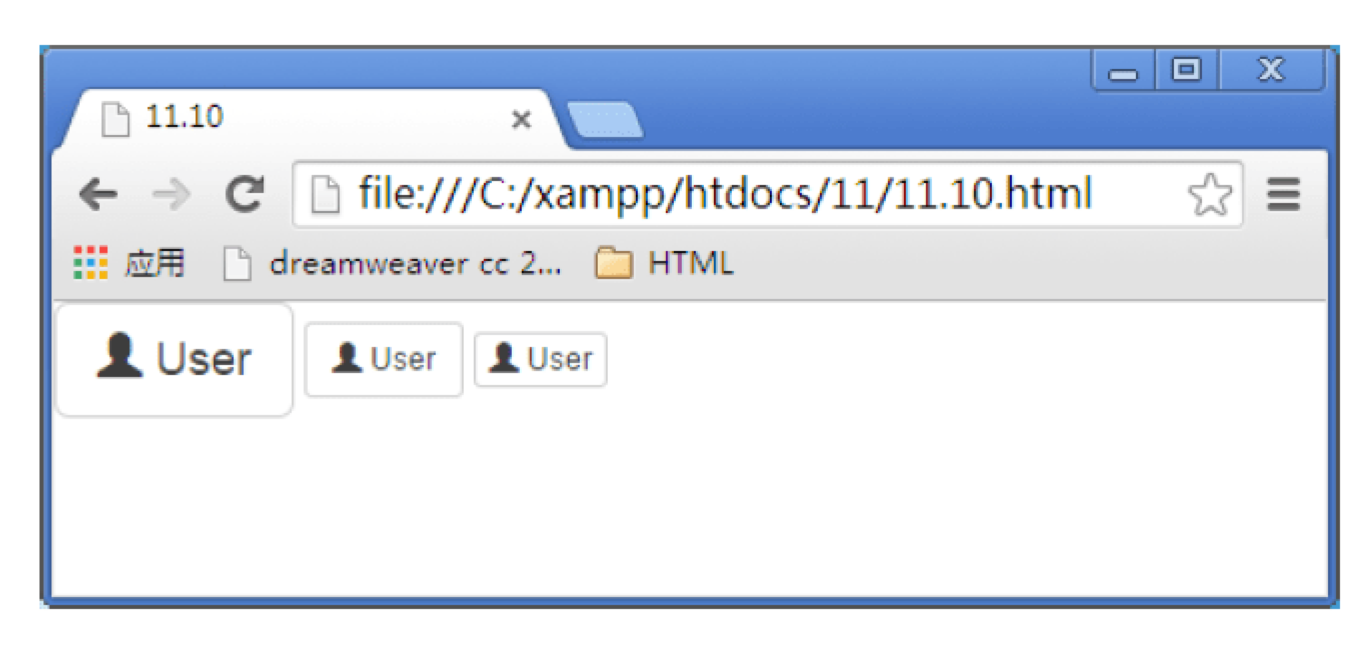


图11.9

关于bootstrap的更多用法，可以访问官网（http://getbootstrap.com）或查找相关资料，由于篇幅的关系，这里只做简单的介绍。

第12章 HTML 5基础

HTML 5是下一代HTML标准，虽然现在HTML 5仍处于完善之中，但是大部分浏览器已经支持某些HTML 5的功能。本章我们将介绍HTML 5的基础知识。



12.1 创建一个HTML 5页面

使用Dreamweaver创建HTML页面的时候，选择文档类型为HTML 5，如图12.1所示。这样就可以创建一个HTML 5页面，在代码视图中可以看到如下所示代码：

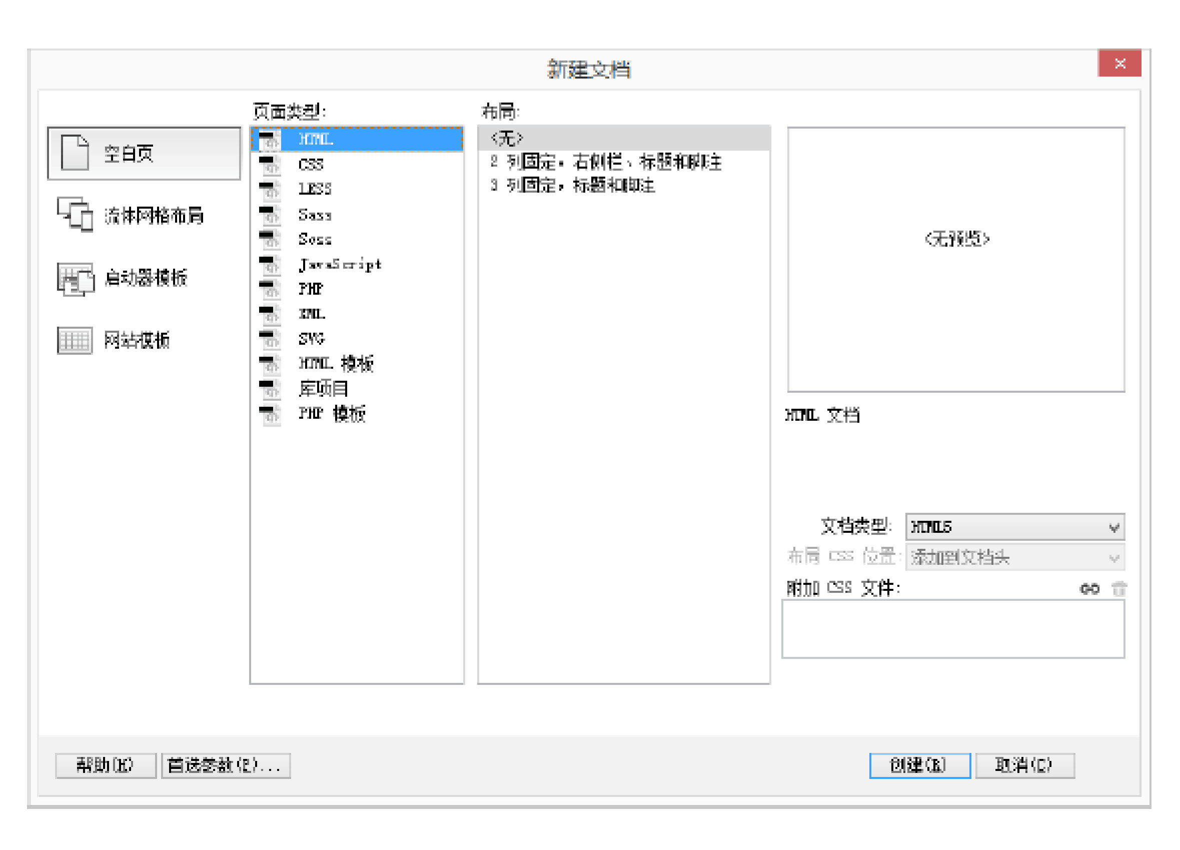


图12.1

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>无标题文档</title>
</head>
<body>
</body>
</html>
```

在这段代码中，文档声明类型非常简单，只有一句代码“<!doctype html>”，而在之前的HTML版本中，文档声明类型如下所示：

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

相比之下，HTML 5使用了更为简单的文档声明方法。这仅仅是HTML 5与之前HTML的一部分差别。在HTML 5中，还使用了体现结构语义化的标签，比如以前用div布局页面时，每一个结构的布局都使用div和id属性来确定。HTML代码如下所示：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>无标题文档</title>
</head>
<body>
<div id="container">
  <div id="header">Header</div>
  <div id="menu">Menu</div>
  <div id="mainContent">
    <div id="sidebar">sidebar</div>
    <div id="content">content</div>
  </div>
  <div id="footer">footer</div>
</div>
</body>
</html>
```

在HTML5中，页面布局可以不再依赖div元素，而是直接使用结构化标签，HTML代码如下所示：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>无标题文档</title>
</head>
<body>
    <header>...</header>
    <nav>...</nav>
    <article>
        <section>
            ...
        </section>
    </article>
    <aside>...</aside>
    <footer>...</footer>
</body>
</html>
```

12.2 HTML 5结构

在HTML 5中采用了语义化的文档结构，新添加了相关的结构元素标签，使用这些标签可以创建新的HTML 5结构。

12.2.1 section标签

section标签用于定义文档中的节，比如章节、页眉、页脚或文档中的其他部分。一般用于成节的内容，会在文档流中开始一个新的节。它用来表现普通的文档内容或应用区块，通常由内容及其标题组成。但section元素标签并非一个普通的容器元素，它表示一段专题性的内容，一般会带有标题。

当我们描述一件具体事物的时候，通常鼓励使用article来代替section；当我们使用section时，仍然可以使用h1元素来作为标题，而不用担心它所处的位置，以及其它地方是否用到；当一个容器需要被直接定义样式或通过脚本定义行为时，推荐使用div元素而非section。

section结构示例代码如下所示：

```
<!doctype html>
<article>
    <h1>Web编程语言比较</h1>
    <p>web编程语言常用的有asp,asp.net,php,jsp...</p>
    <section>
        <h2>asp</h2>
        <p>asp全称Active Server Page</p>
    </section>
    <section>
        <h2>asp.net</h2>
        <p>asp的颠覆版本</p>
    </section>
    <section>
        <h2>php</h2>
        <p>草根动态语言，免费，强大</p>
    </section>
</article>
```

12.2.2 article标签

<article>是一个特殊的section标签，它比section具有更明确的语义，它代表一个独立的、完整的相关内容块，可独立于页面其它内容使用。例如一篇完整的论坛帖子、一篇博客文章、一个用户评论等等。一般来说，article会有标题部分（通常包含在header内），有时也会包含footer。article可以嵌套，内层的article对外层的article标签有隶属关系。例如，一篇博客的文章，可以用article显示，然后一些评论可以以article的形式嵌入其中。

article结构示例代码如下所示：

```
<article>
        <header>
            <hgroup>
                <h1>这是一篇介绍HTML 5结构标签的文章</h1>
                <h2>HTML 5的标签结构</h2>
            </hgroup>
            <time datetime="2015-06-10">2015.06.10</time>
        </header>
        <p>文章内容详情</p>
    </article>
```

12.2.3 nav标签

nav标签代表页面的一个部分，是一个可以作为页面导航的链接组，其中的导航元素链接到其它页面或者当前页面的其它部分，使html代码在语义化方面更加精确，同时对于屏幕阅读器等设备的支持也更好。

nav结构示例代码如下所示：

```
<nav>
 <ul>
  <li>asp</li>
  <li>asp.net</li>
  <li>html 5</li>
 </ul>
</nav>
```

12.2.4 aside标签

aside标签用来装载非正文的内容，被视为页面里面一个单独的部分。它包含的内容与页面的主要内容是分开的，可以被删除，而不会影响到网页的内容、章节或是页面所要传达的信息。例如广告、成组的链接、侧边栏等等。

aside结构示例代码如下所示：

```
<aside>
  <h1>鲁迅</h1>
   <p>鲁迅（1881年9月25日—1936年10月19日），原名周樟寿，后改名周树人，字豫山，后改豫才，"鲁迅"是他1918年发表《狂人日记》时所用的笔名，也是他影响最为广泛的笔名，浙江绍兴人</p>
</aside>
```

12.2.5 header标签

<header>标签定义文档的页眉，通常是一些引导和导航信息。它不局限于写在网页头部，也可以写在网页内容里面。通常<header>标签至少包含（但不局限于）一个标题标记（<h1>-<h6>），可以包括<hgroup>标签，还可以包括表格内容、标识、搜索表单、<nav>导航等。

header结构示例代码如下所示：

```
<header>
 <hgroup>
  <h1>网站标题</h1>
  <h1>网站副标题</h1>
 </hgroup>
</header>
```

12.2.6 footer标签

footer标签定义section或document的页脚，包含了与页面、文章或是部分内容有关的信

息，比如说文章的作者或者日期。作为页面的页脚时，一般包含了版权、相关文件和链接。它和<header>标签使用基本一样，可以在一个页面中多次使用，如果在一个区段的后面加入footer，那么它就相当于该区段的页脚了。

footer结构示例代码如下所示：

```
<footer>
   Copyright &copy2014-2015
</footer>
```

12.2.7 hgroup标签

hgroup标签是对网页或区段section的标题元素（h1-h6）进行组合。例如，在一区段中有连续的h系列的标签元素，就可以用hgroup将他们括起来。

hgroup结构示例代码如下所示：

```
<hgroup>
  <h1>大标题</h1>
  <h2>小标题</h2>
</hgroup>
```

12.2.8 figure标签

figure标签用于对元素进行组合，多用于图片与图片描述组合。figure结构示例代码如下所示：

```
<figure>
  <img src="img01.gif" />
  <figcaption>图片描述信息</figcaption>
</figure>
```

第13章 HTML 5音频与视频

在HTML 5之前，为了让网页具有音频和视频的效果，往往需要使用第三方插件或者flash工具；而HTML 5提供了音频和视频的播放接口，新增了video和audio标签，所有支持HTML 5的浏览器都可以直接播放音频和视频。



13.1 检查浏览器是否支持HTML 5的功能

由于目前并非所有的浏览器都很好地支持HTML 5的功能，所以为了避免部分浏览器不支持某些属性的情况出现，在使用HTML 5的音频和视频之前，需要对浏览器的支持情况进行检测。运行下面这段代码，单击“检测”按钮即可查看当前浏览器是否支持HTML 5的视频播放功能。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<script type="text/javascript">
function checkVideo(){
if(!!document.createElement('video').canPlayType)  {
  var vidTest=document.createElement("video");
  oggTest=vidTest.canPlayType('video/ogg; codecs="theora, vorbis"');
  if (!oggTest)    {
      h264Test=vidTest.canPlayType('video/mp4; codecs="avc1.42E01E, mp4a.40.2"');
    if (!h264Test)      {
      document.getElementById("checkVideoResult").innerHTML="您的浏览器不支持HTML 5视频播放！"
      }
    else {
      if (h264Test=="probably") {
         document.getElementById("checkVideoResult").innerHTML="您的浏览器支持HTML 5视频播放！";
        }
      else {
         document.getElementById("checkVideoResult").innerHTML="您的浏览器支持部分HTML 5视频播放！";
        }
      }
```

```
        }
      else {
        if (oggTest=="probably") {
           document.getElementById("checkVideoResult").innerHTML="您的浏览器支持
HTML 5视频播放！";
          }
        else {
          document.getElementById("checkVideoResult").innerHTML="您的浏览器支持部
分HTML 5视频播放！";
          }
        }
      }
    else {
      document.getElementById("checkVideoResult").innerHTML="您的浏览器不支持HTML
5视频播放！"
      }
    }
    </script>
    <title>13.1</title>
    </head>
    <body>
    <div id="checkVideoResult" style="margin:10px 0 0 0; border:0;
padding:0;">
    <button onclick="checkVideo()" style="font-family:Arial, Helvetica, sans-
serif;">检测</button>
    </div>
    </body>
    </html>
```

然而在实际使用中，并非每次都需要执行以上代码才能判断浏览器是否支持HTML 5的视频功能，在页面中执行以下代码也可以进行判断。

```
    <script>
    var hasVideo = !!(document.createElement('video').canPlayType);
    alert(hasVideo);
    </script>
```



13.2 添加音频和视频文件

在HTML 5中，可以使用video和audio标签添加音频和视频文件。由于目前video元素支持Ogg、MPEG4和WebM格式的视频文件，而audio元素支持Ogg Vorbis、MP3和Wav格式的音频文件，而且不同的浏览器支持音频和视频的程度也不一样，所以在添加音视频文件的时候需要特别注意。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>13.2</title>
</head>
<body>
<audio controls src="Source/song.mp3"></audio>
<video controls src="Source/movie.mp4"></video>
</body>
</html>
```

在这段代码中，<audio>元素和<video>元素分别使用src指定了一个音频和视频文件，而controls属性用于告诉浏览器显示通用的用户控制，包括开始、停止、调播以及音量控制。如果不指定controls特性，用户将无法播放页面上的音频和视频。如图13.1所示是谷歌浏览器中音频和视频播放控件的效果。

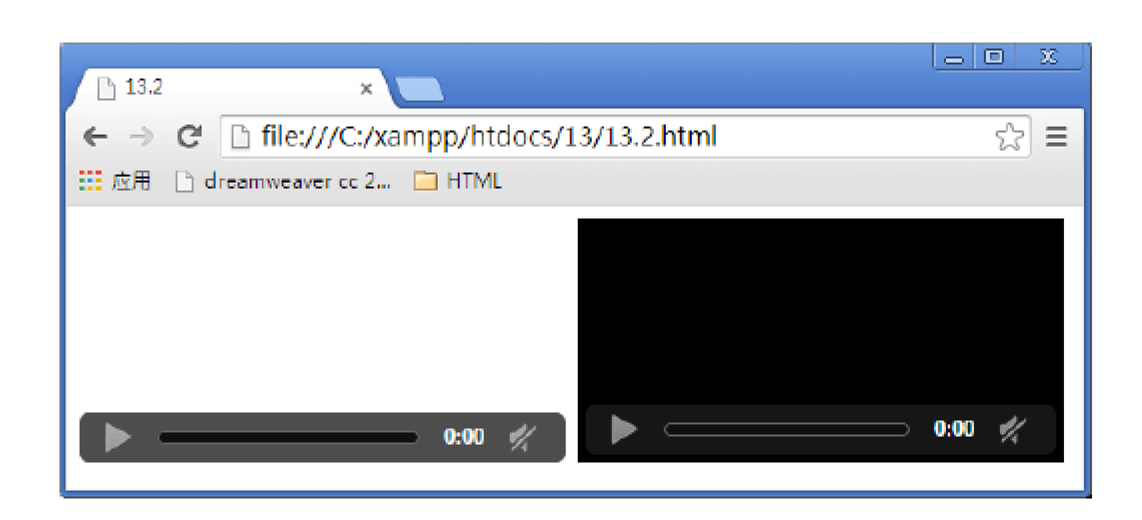


图13.1



13.3 指定备用的文件源

<audio>元素和<video>元素中的src属性用于指定音频和视频的数据源地址，目前这两个元素都支持3种格式的的音频和视频文件，所以最多可以有3个数据源。当浏览器不支持播放第1种格式文件的时候，它会自动选择播放第2种格式文件，如果第2种也不支持，它会尝试继续下一种格式。

如果要指定多个数据源，就需要用到source标签，相关代码如下所示：

```
<audio controls>
        <source src="Source/song01.ogg" />
        <source src="Source/song02.mp3" />
</audio>
<video controls>
        <source src="Source/mv01.ogg" />
        <source src="Source/mv02.mpeg4" />
 </ video >
```

13.4 video和audio元素的属性

我们上面介绍了video和audio元素的src和controls属性，为了方便控制音频和视频播放，video和audio元素还提供了其他很多属性。

1. autoplay属性

当页面加载时，设置是否自动播放音频和视频文件。如果需要自动播放音频和视频文件，则添加该属性，否则不添加该属性。相关使用代码如下：

```
<video src="movie.mp4" autoplay ></video>
<audio src="song.ogg" autoplay ></ audio>
```

2. preload属性

当页面加载时，设置是否对音频和视频文件进行预加载。preload属性有3个可供选择的值，none表示不进行预加载；metadata表示仅加载元数据，即音频和视频文件的大小、第一帧、播放列表和持续时间等；auto表示预加载全部音频和视频文件。相关使用代码如下：

```
<video src="movie.mp4" preload="metadata"></video>
<audiosrc="song.ogg" preload="metadata">
```

3. poster属性

该属性是video元素属性，设置当视频不可用时，向用户展现一幅图片。相关使用代码如下：

```
<video src="movie.mp4" poster="replace.jpg "></video>
```

4. loop属性

设置是否循环播放音频和视频文件。如果需要循环播放音频和视频文件，则添加该属性，否则不添加该属性。相关使用代码如下：

```
<video src="movie.mp4" loop ></video>
<audio src="song.ogg" loop ></ audio>
```

5. width属性和height属性

这两个属性是video元素属性，width属性用于指定视频的宽度，height属性用于指定视频的高度，单位均是像素。相关使用代码如下：

```
<video src="movie.mp4" width="400" height="300" ></video>
```

6. muted属性

当页面加载时，设置播放器是否被静音。如果需要静音，则添加该属性，否则不添加该属性。相关使用代码如下：

```
<video src="movie.mp4" muted></video>
<audio src="song.mp3" muted ></audio>
```

 13.5 使用JavaScript控制播放

使用JavaScript也可以控制video和audio元素进行播放。在HTML 5中提供了4种控制播放功能的方法，分别介绍如下：

1. play方法

除了播放器自己的播放功能外，用户还可以在脚本中使用play方法来控制音频和视频的播放功能，相关使用代码如下：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>13.3</title>
<script>
function play(){
var video=document.getElementById("MyVideo");
video.play();
}
</script>
</head>
<body>
<video id="MyVideo" src="Source/movie.mp4" controls></video>
<button onClick="play()">play</button>
</body>
</html>
```

2. pause方法

与play方法相对应的pause方法用于设置暂停播放音频和视频功能，pause方法也需要在脚本中设置才能使用，相关使用代码如下：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>13.4</title>
<script>
function pause(){
```

```
var video=document.getElementById("MyVideo");
video.pause();
}
</script>
</head>
<body>
<video id="MyVideo" src="Source/movie.mp4" controls></video>
<button onClick="pause()">pause</button>
</body>
</html>
```

3. load方法

调用该方法可重新加载音频和视频文件进行播放。相关使用代码如下：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>13.5</title>
<script>
function load(){
var video=document.getElementById("MyVideo");
video.load();
}
</script>
</head>
<body>
<video id="MyVideo" src=" Source/movie.mp4 " controls></video>
<button onClick="load()">load</button>
</body>
</html>
```

4. canPlayType方法

该方法用于测试浏览器是否支持指定的类型，并返回结果。如果返回空字符串则表示浏览器不支持此种播放类型，如果返回maybe则表示浏览器可能支持此种播放类型，如果返回probably则表示浏览器确定支持此种播放类型，详见13.1章节。



13.6 音频和视频播放事件

在音频和视频播放的整个过程中，会触发一系列的事件，捕获这些事件并加以利用，就可以实现更多的效果。在HTML 5中，与video和audio元素相关的事件如表13.1所示。

表13.1

属性	描述
loadstart	浏览器开始在网上寻找媒体数据
progress	浏览器正在获取媒体数据
suspend	浏览器暂停获取媒体数据，但是下载过程并没有正常结束
abort	浏览器在下载完全部媒体数据之前终止获取媒体数据，但是并不是由错误引起的
error	获取媒体数据过程中出错
emptied	video元素或audio元素所在网络突然变为未初始化状态（可能引起该事件的原因有两个：1. 载入媒体时突然发生一个致命错误；2. 在浏览器正在选择支持的播放格式时，又调用了load方法重新载入媒体。）
stalled	浏览器尝试获取媒体数据失败
play	即将开始播放，当执行play方法时触发，或数据下载后元素被设置为autoplay属性
pause	播放暂停，当执行pause方法时触发
loadedmetadata	浏览器获取完毕媒体的时间长和字节数
loadeddata	浏览器已加载完毕到当前播放为止的媒体数据，准备播放
waiting	播放过程由于得不到下一帧而暂停播放（例如下一帧尚未加载完毕），但很快就能够得到下一帧
playing	正在播放
canplay	浏览器能够播放媒体，但估计以当前播放速率不能直接将媒体播放完毕，播放期间需要缓冲
canplaythrough	浏览器能够播放媒体，而且以当前播放速率能够将媒体播放完毕，不再需要进行缓冲
seeking	seeking属性变为true，浏览器正在请求数据
seeked	seeked属性变为false，浏览器停止请求数据
timeupdate	当前播放被改变，可能是播放过程中的自然改变，也可能是被人为改变，或由于播放不能连续而发生的跳变
ended	播放结束后停止播放
rantechange	defaultplaybackrate属性（默认播放速率）或playbackrate属性（当前播放速率）被改变
durationchange	播放时长被改变
volumechange	volume属性（音量）被改变或muted属性（静音状态）被改变

在JavaScript中捕捉事件的方式有两种，第1种是监听的方式，使用video元素或audio元素的addEventListener方法对事件进行监听。相关代码如下：

```
var video=document.getElementById("MyVideo");
var img=document.getElementById("MyImg");
video.addEventListener("play",function(){
img.hidden=true;
},false);
```

video表示页面上的video元素或audio元素，play表示监听的事件名称，在function中处理当音频和视频播放的时候需要执行的其他操作，img.hidden=true表示页面上的一个图片被隐藏，false表示浏览器采用bubbing响应方式，如果为true则表示浏览器采用Capture响应方式。

另一种捕捉事件的方式是在JavaScript脚本中获取事件句柄（事件名称前加on就是事件句柄），并对事件句柄赋值。相关代码如下：

```
    var img=document.getElementById("MyImg");
    function showImg(flag){
    img.hidden=flag;
    }
    <video id="MyVideo" src="movie.mp4" controls onPlay="showImg(true);"
onPause="showImg(false);" ></video>
    <img id="MyImg" src="myimage.png" hidden="true"/>
```

第14章 CSS 3使用指南

为了适应Web 2.0技术，CSS 3推出了新的标准，并且提供了一系列强大的功能，比如许多新的CSS属性、各种CSS特效，甚至还有CSS动画、元素的变换。本章将详细介绍CSS 3的新特性以及一些使用技巧。



14.1 CSS 3选择器

CSS 3中新添加了很多新的选择器，使用这些选择器可以更准确、更方便地定位HTML页面中的元素。

14.1.1 结构性伪类选择器

结构性伪类选择器与类选择器有很大的不同，它不需要用户指定元素的类名，而是由CSS提供选择器名称，从而快速定位指定的元素。在CSS 3中，结构性伪类选择器还可以细分为以下几种：

1. 伪元素选择器

伪元素选择器主要应用于指定名称的元素上，不需要用户单独为其设置选择器名称，而是使用CSS 3中已经定义的选择器名称，这类选择器主要包括以下4种。

（1）first-line伪元素选择器：为元素中第一行的文字应用样式。
（2）first-letter伪元素选择器：为元素中第一个字母或文字应用样式。
（3）before伪元素选择器：在指定元素之前插入一些内容。
（4）after伪元素选择器：在指定元素之后插入一些内容。

例如下面这段代码，在<p>元素中有4行文本，使用伪元素选择器设置第一行文本的颜色为红色，第一个文字大字号为24px，在<span>标签的前面插入一个三角形符号，在<p>元素的后面插入一个五角星符号，完整的代码如下所示：

```
<!doctype html>
<html>
<head>
```

```
<meta charset="utf-8">
<title>14.1</title>
<style>
p:first-line {
color: red;
}
p:first-letter {
font-size: 24px;
}
span:before {
content: "△"
}
p:after {
content: "☆"
}
</style>
</head>
<body>
<p>床前明月光<br>
  疑是地上霜<br>
  <span>举头望明月</span><br>
  低头思故乡</p>
</body>
</html>
```

运行这段代码后，效果如图14.1所示。

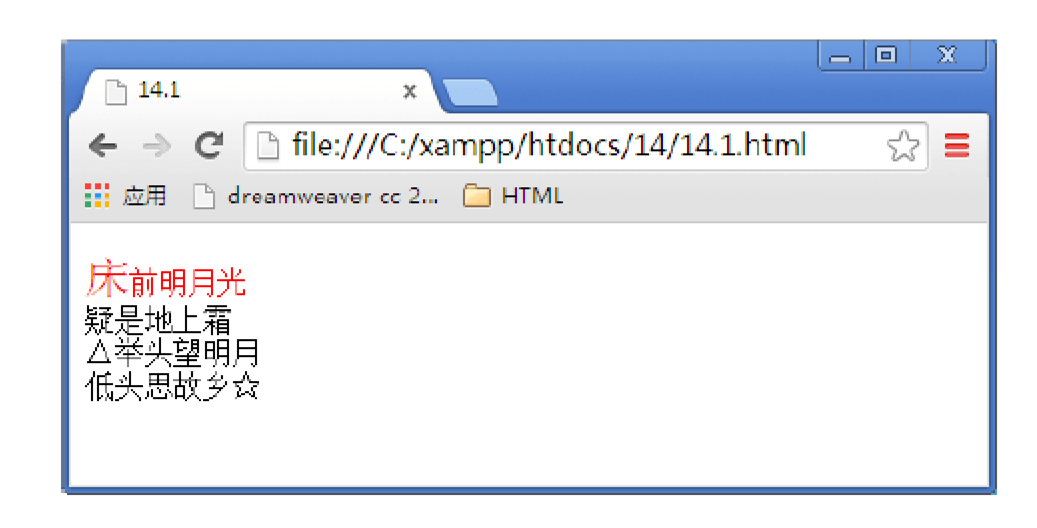


图14.1

2. root选择器

root选择器用于设置页面根元素的样式。在HTML页面中，根元素就是位于最顶层结构的<html>元素，它的使用方法如下所示：

```
<style>
:root{
background:blue;
}
</style>
```

3. not选择器

HTML页面中的各种元素都是嵌套在一起使用的，当用户为上一级元素设置样式后，由于样式的继承特性，被嵌套的元素会具有相同的样式。为了能够排除嵌套元素中某些不使用

继承样式的元素，可以使用not选择器将其排除。例如，<div>元素中嵌套了多个元素，我们设置<div>元素字体颜色为红色，但是不设置<p>元素的颜色，可以使用以下代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.2</title>
<style>
div :not(p){
color:red;
}
</style>
</head>
<body>
<div>
<h1>标题</h1>
<p>文章的内容</p>
</div>
</body>
</html>
```

运行这段代码后，效果如图14.2所示。

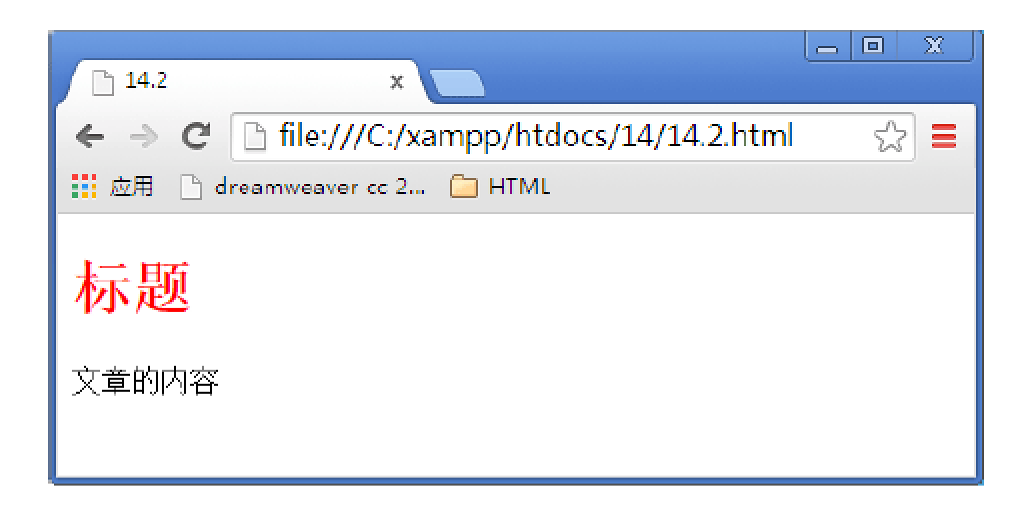


图14.2

4. empty选择器

如果要为HTML页面中所有内容为空的元素设置样式，需要使用JavaScript获取所有内容为空的元素，然后为其设置样式。但是在CSS 3中，可以直接使用empty选择器很方便地完成这个功能。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.3</title>
<style>
:empty{
width:50px;
height:20px;
background-color:red;
}
```

```
</style>
</head>
<body>
<p></p>
<p>CSS 3新增的各种选择器</p>
</body>
</html>
```

在这段代码中，HTML页面中有两个<p>元素，其中第一个<p>元素的内容为空，为了凸显该元素，直接使用empty选择器为其设置样式。运行这段代码后，效果如图14.3所示。

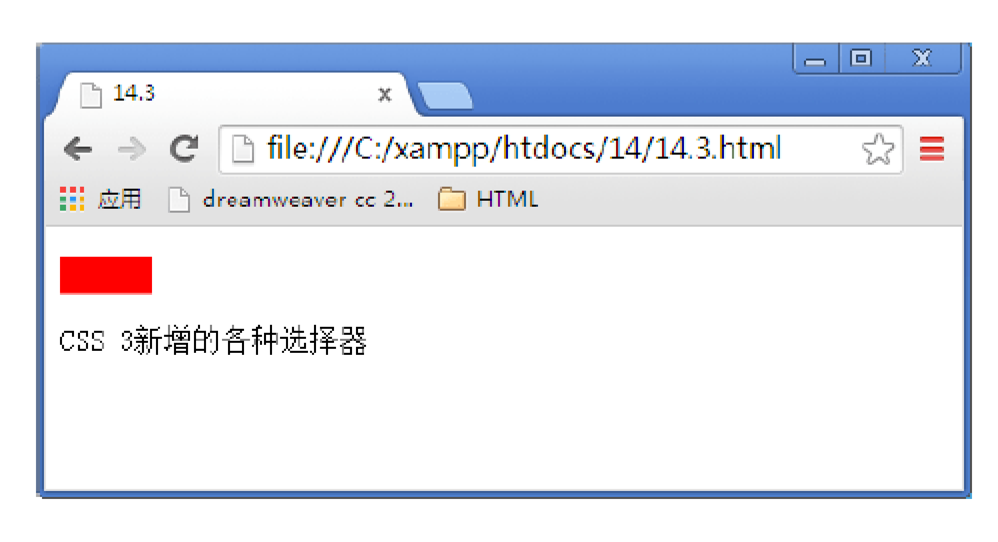


图14.3

5. target选择器

target选择器用于指定页面中某个元素的样式。在使用target选择器时，需要根据元素的id属性指定样式，而元素的id又被当做页面中的超链接来使用，这样target选择器的样式才能起作用。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.4</title>
<style>
:target{
background:red;
font-size:24px;
color:white;
}
</style>
</head>
<body>
<a href="#p1">第一个段落</a>
<a href="#p2">第二个段落</a>
<p id="p1">这里显示的是第一个段落。</p>
<p id="p2">这里显示的是第二个段落。</p>
</body>
</html>
```

在这段代码中有两个<p>元素，他们的id属性分别对应另外两个<a>元素的锚点，当点击超链接时，target选择器会根据选中元素的锚点找到对应id的<p>元素，并将设置的样式应用

到该元素上，效果如图14.4所示。

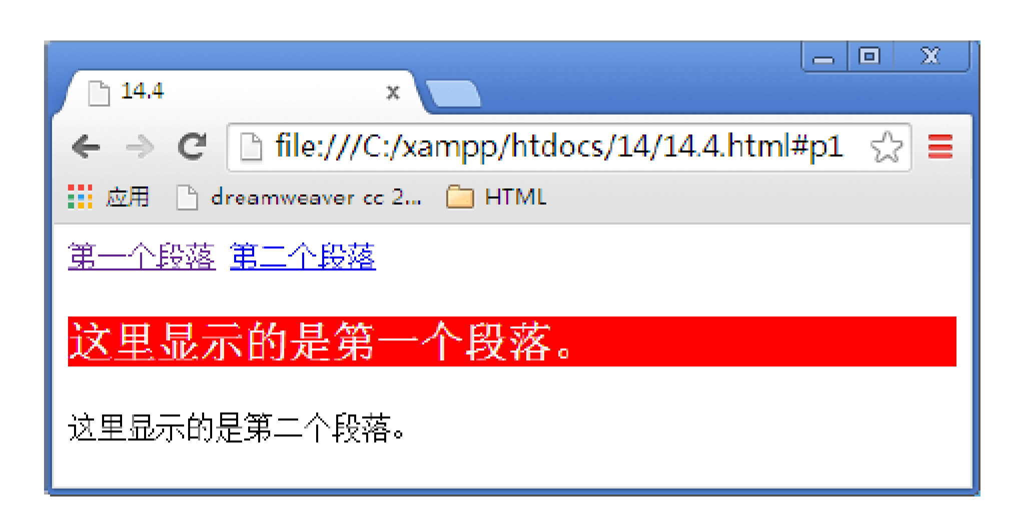


图14.4

7. first-child和last-child选择器

在使用列表时，有时需要为第一个列表项和最后一个列表项使用不同的样式，为了定位这两个元素，就需要为它们设置不同的id属性。然而在CSS 3中，可以省略id属性，直接使用first-child和last-child选择器定位这两个元素。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.5</title>
<style>
li:first-child{
color:red;
font-weight:bold;
}
li:last-child{
color:blue;
font-weight:bold;
}
</style>
</head>
<body>
<ul>
<li>列表项</li>
    <li>列表项</li>
    <li>列表项</li>
    <li>列表项</li>
    <li>列表项</li>
</ul>
</body>
</html>
```

在这段代码中，分别使用first-child和last-child选择器为第一个列表项和最后一个列表项设置了不同的样式，效果如图14.5所示。

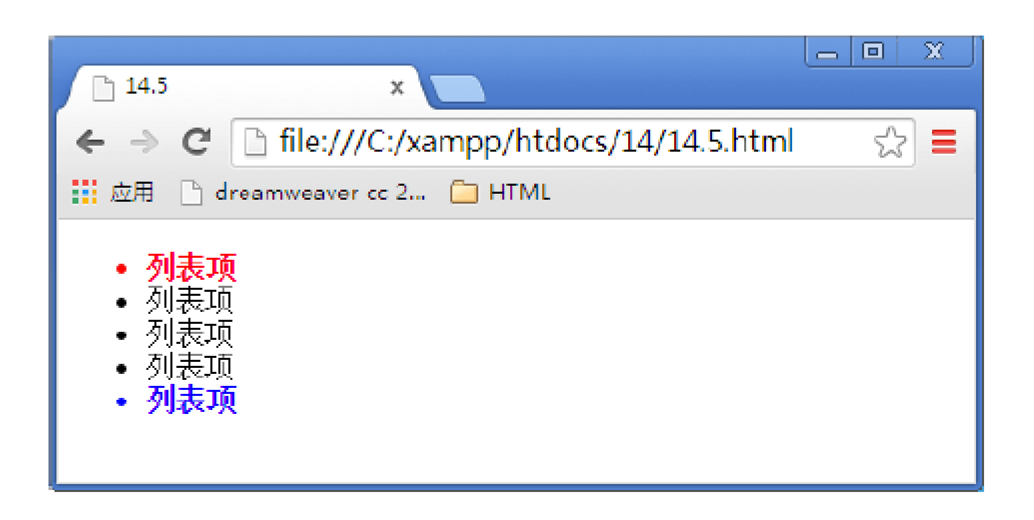


图14.5

8. first-of-type和last-of-type选择器

在使用first-child和last-child选择器时，如果相同层级中有多种元素，我们只能设置第一种元素和最后一种元素的样式，不能设置第一个和最后一个相同元素的样式。为了满足这样的需求，可以使用first-of-type和last-of-type选择器定位相同类型的第一个元素和最后一个元素。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.6</title>
<style>
.content:first-of-type {
background: red;
font-size: 24px;
color: white;
}
.content:last-of-type {
background: green;
font-size: 24px;
color: white;
}
</style>
</head>
<body>
<div>
  <h1 class="content">标题1</h1>
  <p class="content">第一个段落。</p>
  <h1 class="content">标题2</h1>
  <p class="content">第二个段落。</p>
  <h1 class="content">标题3</h1>
  <p class="content">第三个段落。</p>
  <h1 class="content">标题4</h1>
  <p class="content">第四个段落。</p>
</div>
</body>
</html>
```

在这段代码中，分别使用first-of-type和last-of-type选择器定位到了第一个和最后一个

<h1>元素，以及第一个和最后一个<p>元素，并为其设置了不同样式。运行这段代码后，效果如图14.6所示。



图14.6

9. nth-child和nth-last-child选择器

nth-child选择器和nth-last-child选择器是对first-child选择器和last-child选择器的扩展。nth-child选择器用于指定父元素中第几个元素的样式，而nth-last-child选择器正好相反，用于指定父元素中倒数第几个元素的样式。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.7</title>
<style>
li:nth-child(2){
color:red;
font-weight:bold;
}
li:nth-last-child(2){
color:blue;
font-weight:bold;
}
</style>
</head>
<body>
<ul>
    <li>列表项1</li>
    <li>列表项2</li>
    <li>列表项3</li>
    <li>列表项4</li>
    <li>列表项5</li>
</ul>
</body>
```

```
</html>
```

在这段代码中，分别使用nth-child选择器和nth-last-child选择器设置第2个列表项和倒数第2个列表项的样式。运行这段代码后，效果如图14.7所示。

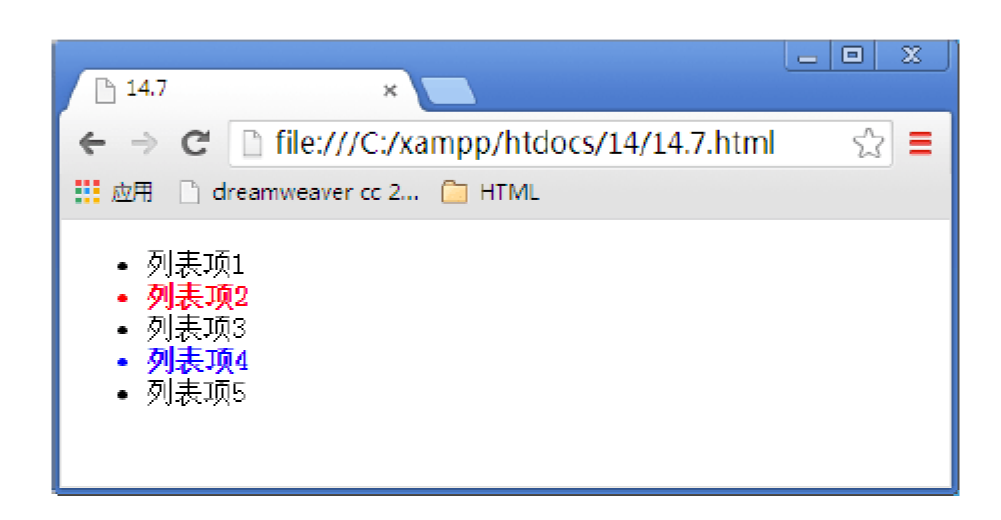


图14.7

10. nth-of-type和nth-last-of-type选择器

与nth-child和nth-last-child选择器对应的，可以使用nth-of-type和nth-last-of-type选择器在定位元素的时候按照元素类型进行定位。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.8</title>
<style>
.content:nth-of-type(odd) {
background: red;
color: white;
}
.content:nth-of-type(even) {
background: green;
color: white;
}
</style>
</head>
<body>
<div>
  <h1 class="content">标题1</h1>
  <p class="content">第一个段落。</p>
  <h1 class="content">标题2</h1>
  <p class="content">第二个段落。</p>
  <h1 class="content">标题3</h1>
  <p class="content">第三个段落。</p>
  <h1 class="content">标题4</h1>
  <p class="content">第四个段落。</p>
</div>
</body>
</html>
```

在这段代码中，分别使用nth-of-type和nth-last-of-type选择器定位<h1>和<p>元素，其中

odd代表奇数，even代表偶数。运行这段代码后，效果如图14.8所示。



图14.8

11. 循环使用式样

对于以上介绍的几种选择器，都需要明确定位到第几个元素，如果需要对某几个元素循环使用样式，那么应该如何设置呢？对于这样的需求，在CSS 3中仍然可以轻松地完成。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.9</title>
<style>
li:nth-child(3n+1) {
background: red;
color: white;
}
li:nth-child(3n+2) {
background: blue;
color: white;
}
li:nth-child(3n+3) {
background: green;
color: white;
margin-bottom:5px;
}
</style>
</head>
<body>
<ul>
<li>列表项A</li>
    <li>列表项B</li>
    <li>列表项C</li>
    <li>列表项A</li>
    <li>列表项B</li>
```

```
        <li>列表项C</li>
        <li>列表项A</li>
        <li>列表项B</li>
        <li>列表项C</li>
    </ul>
    </body>
    </html>
```

在这段代码中，3个nth-child选择器的参数分别为“3n+1”“3n+2”和“3n+3”，其中n前面的3表示总共有3种样式，而1、2、3分别对应循环中的第1种、第2种和第3种样式，也就是为循环中的第几个元素设置样式。运行这段代码后，效果如图14.9所示。

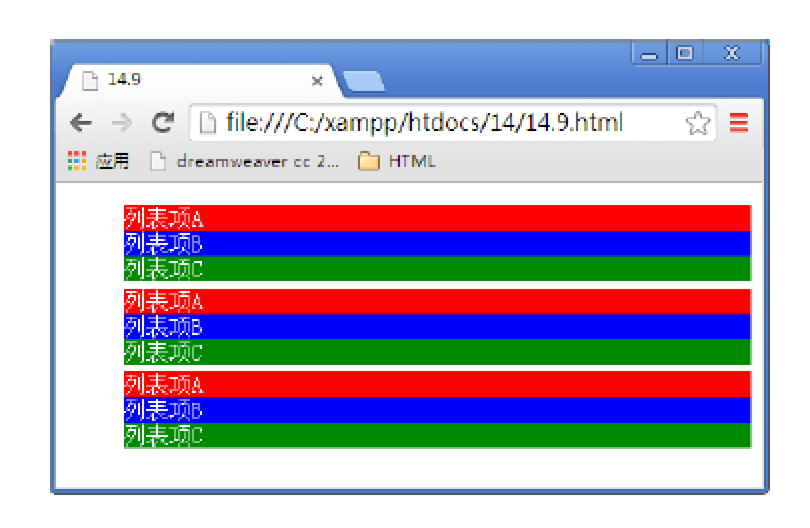


图14.9

12. only-child和only-of-type选择器

only-child选择器是父元素只有一个子元素时使用的样式，而only-of-type选择器是对only-child选择器的扩展，在应用样式时考虑了子元素的类型。例如下面这段代码：

```
    <!doctype html>
    <html>
    <head>
    <meta charset="utf-8">
    <title>14.10</title>
    <style>
    li:only-child {
    color: red;
    font-weight:bold;
    }
    </style>
    </head>
    <body>
    <ul>
    <li>列表项</li>
    </ul>
    <ul>
    <li>列表项</li>
        <li>列表项</li>
        <li>列表项</li>
    </ul>
    </body>
    </html>
```

在这段代码中，页面中有两个列表，第1个列表只有1个列表项，第2个列表中有3个列表项，使用only-child选择器设置第1个列表的列表项样式。运行这段代码后，效果如图14.10所示。

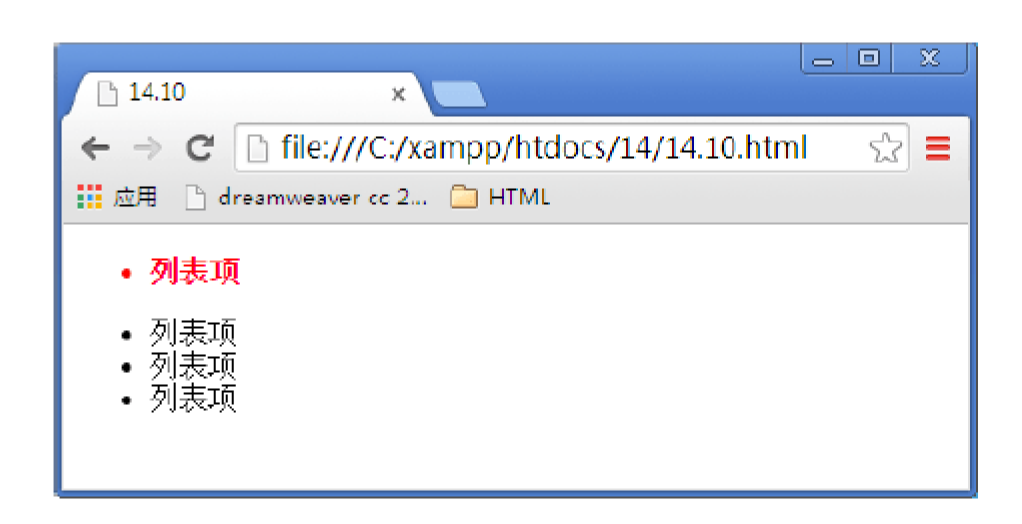


图14.10

14.1.2 UI元素状态伪类选择器

从字面意思理解，这类选择器只有当元素处于某种状态的时候，才能通过对应的选择器定位到元素。例如，经常用到的E:link、E:visited、E:hover和E:active就属于这类选择器。在CSS 3中，又新增了E:enabled、E:disabled和E:checked选择器，E:enabled选择器用于选择所有可用的元素，E:disabled选择器用于选择所有不可用的元素，E:checked选择器用于选择所有选中的元素。

14.1.3 通用兄弟元素选择器

通用兄弟元素选择器用于设置父级元素相同、子元素有多种类型、从某个类型的子元素往后的其他子元素的样式。例如下面的代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.11</title>
<style>
h3~p {
background: red;
color: white;
}
</style>
</head>
<body>
<div>
    <h3>标题</h3>
    <p>这里是内容</p>
    <ul>
        <li><p>列表项目</p></li>
        <li><p>列表项目</p></li>
    </ul>
</div>
</body>
</html>
```

在这段代码中，<h3>元素和相邻的<p>元素属于同一级，可以称之为兄弟元素。所以这里使用h3~p选择器直接定位这个<p>元素并设置样式，但是这并不影响列表中<p>元素的样式。运行这段代码后，效果如图14.11所示。

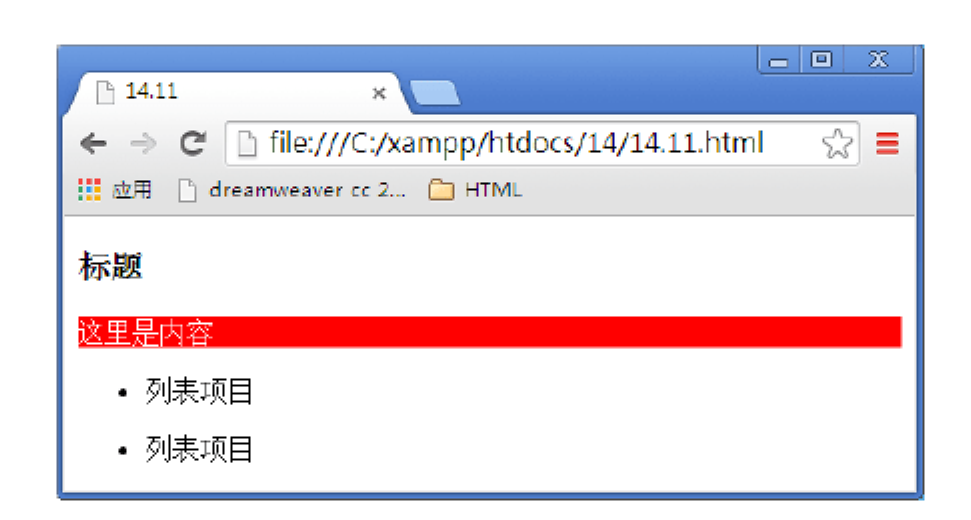


图14.11

 14.2 @Font-face特性

@font-face可以用来加载字体样式，而且它还能够加载服务端的字体文件，即便客户端没有安装相应的字体，也可以通过这种方式加载正确的字体。例如下面这段代码：

```
<style>
@font-face{
font-family:BorderWeb;
src:url(BORDERW0.eot);
}
@font-face {
font-family: Runic;
src:url(RUNICMT0.eot);
 }
 .border { font-size:35px; color:black; font-family:"BorderWeb" }
 .event { font-size:24px; color:black; font-family:"Runic" }
</style>
```

在这段代码中，首先使用@font-face声明两个服务端字体，这两个服务端字体名称分别为BorderWeb和Runic，并分别指定其来源为BORDERW0.eot和RUNICMT0.eot。这样我们就可以在样式中直接使用这两个字体了，即使客户端上没有安装这两种字体，仍然可以正确显示。

 14.3 Word-wrap和Text-overflow

Word-wrap属性用于设置长单词换行到下一行，例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.13</title>
<style>
p:first-child{
width:300px;
border:1px solid #832A2C;
}
p:last-child{
width:300px;
border:1px solid #832A2C;
word-wrap:break-word;
}
</style>
</head>
<body>
<p>asdfjlasldfjlasdfllkajsdfkljasjdflkjsdajkflasdjkfasljfd</p>
<p>asdfjlasldfjlasdfllkajsdfkljasjdflkjsdajkflasdjkfasljfd</p>
</body>
</html>
```

在这段代码中有两个相同的段落文本，第1个段落没有设置word-wrap属性，第2个段落设置了word-wrap属性。运行这段代码后，效果如图14.12所示。

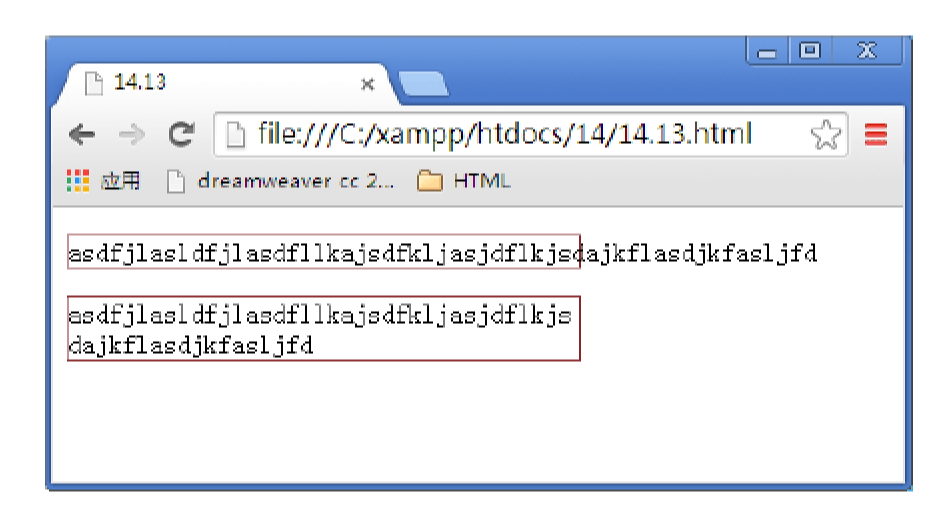


图14.12

Text-overflow属性用于设置当容器无法容纳文本时应该如何进行处理。需要注意的是，该属性必须和overflow:hidden配合使用，否则没有效果。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.14</title>
<style>
p:first-child{
width:300px;
border:1px solid #832A2C;
}
p:last-child{
width:300px;
```

```
border:1px solid #832A2C;
overflow:hidden;
text-overflow:clip;
}
p:nth-child(2){
width:300px;
border:1px solid #832A2C;
overflow:hidden;
text-overflow:ellipsis;
}
</style>
</head>
<body>
<p>asdfjlasldfjlasdfllkajsdfkljasjdflkjsdajkflasdjkfasljfd</p>
<p>asdfjlasldfjlasdfllkajsdfkljasjdflkjsdajkflasdjkfasljfd</p>
<p>asdfjlasldfjlasdfllkajsdfkljasjdflkjsdajkflasdjkfasljfd</p>
</body>
</html>
```

在这段代码中，有3段相同的文本，分别为这3段文本设置不同的样式，其中第2段文本和第3段文本都设置了不同的text-overflow属性。运行这段代码后，效果如图14.13所示。

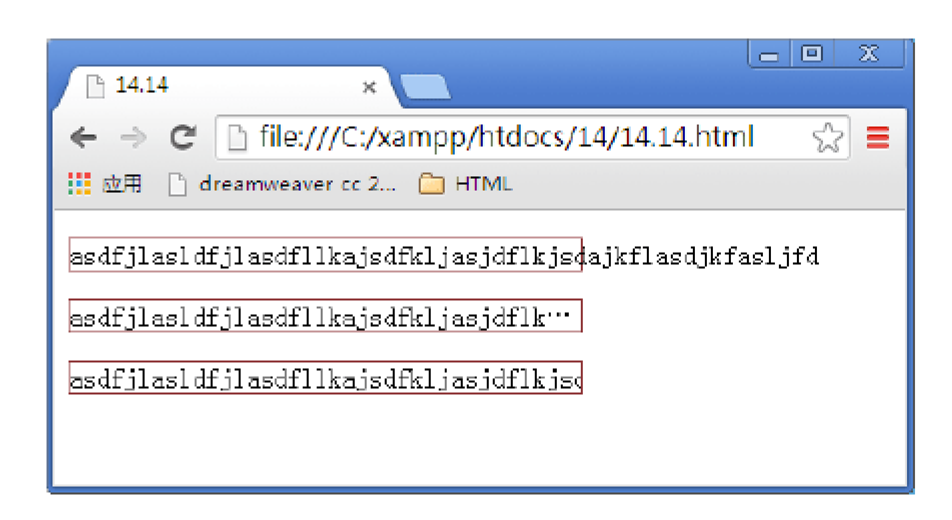


图14.13



14.4 CSS 3的多列布局

我们经常可以在报纸上看到一个页面排版多列的布局方式，现在CSS 3中也可以轻松实现这样的布局。要实现这样的效果，需要使用以下3个属性。

- Column-count：表示布局几列。
- Columne-rule：表示列与列之间的间隔条的样式。
- Column-gap：表示列与列之间的间隔。

例如下面这段代码：

```
<!doctype html>
```

```
<html>
<head>
<meta charset="utf-8">
<title>14.15</title>
<style>
p {
-webkit-column-count: 3;
-webkit-column-rule: 1px solid #bbb;
-webkit-column-gap: 2em;
 }
</style>
</head>
<body>
<p>它曾经是，在1888年以前。1888年北洋舰队正式成军，此时它有25艘舰艇，旅顺、威海、大沽三处主要基地，军队训练水平不亚于西方的远东海上劲旅，其中"定远"、"镇远"2艘一等铁甲舰称雄亚洲。然而清政府不懂得海军是一个需要连续投入、不断更新的军种，此后就紧缩开支，甚至严令禁止再添购新的舰、炮、军火，最终北洋海军的主力只剩下8艘军舰。</p>
</body>
</html>
```

在这段代码中，使用-webkit-column-count设置3列布局，使用-webkit-column-rule设置列之间的间隔样式，使用-webkit-column-gap设置列间隔为2em。运行这段代码后，效果如图14.14所示。

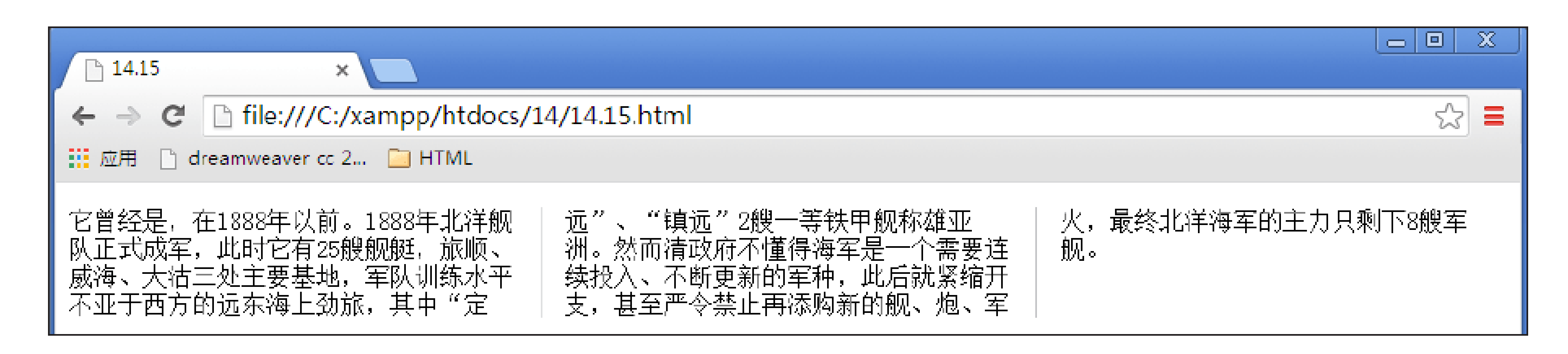


图14.14



14.5 边框和颜色

以前为了在页面中制作一个圆角效果，总是需要费尽周折，而现在CSS 3中提供了圆角边框的设置，通过border-radius属性可以直接设置圆角边框。例如，使用下面的代码可以直接设置圆角半径为30px的div，效果如图14.15所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.16</title>
```

```
<style>
div{
width:200px;
height:160px;
border-radius:30px;
background-color:#A38484;
}
</style>
</head>
<body>
<div></div>
</body>
</html>
```

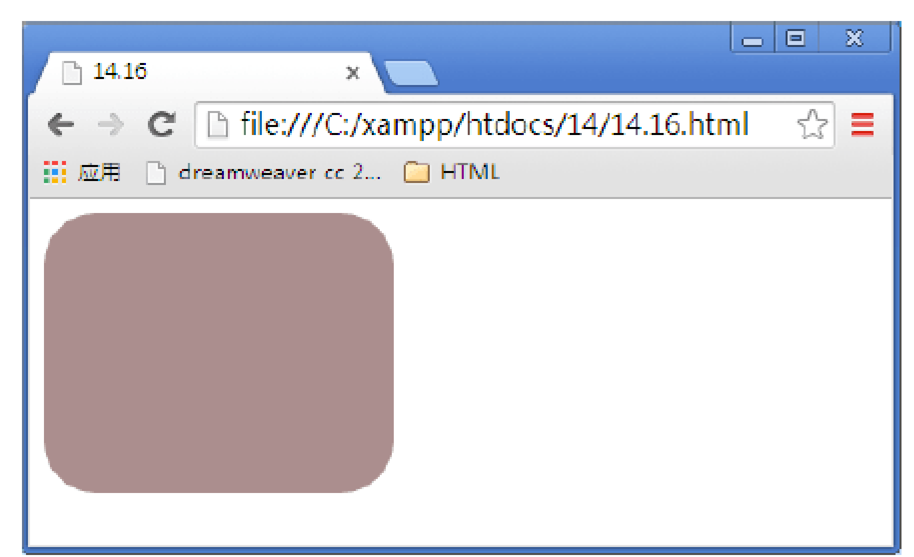


图14.15

CSS 3还提供了两种对颜色透明度支持的方式，一种是通过rgba来设置透明度，另一种是通过hsl来设置透明度。相关代码如下所示，这里的0.75和0.68就代表了透明度。

```
background: rgba(0, 0, 255, 0.75);
color: hsla( 112, 72%, 33%, 0.68);
```

14.6 CSS 3的渐变效果

在为某些元素设置背景的时候，经常会用到渐变效果，为了达到这种效果，我们会使用一幅有渐变效果的图片作为背景。在CSS 3中就不需要这么麻烦了，只需要通过属性设置即可完成渐变效果。

14.6.1 线性渐变

线性渐变可以实现从一点向另一点的颜色过渡，可以设置一个起点和一个方向，或者是一个角度。线性渐变的标准语法如下所示：

```
background: linear-gradient(direction, color-stop1, color-stop2, ...);
```

direction表示渐变的方向，color-stop表示渐变的颜色，线性渐变至少需要两种以上颜色才能实现。线性渐变默认从上到下渐变，例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.17</title>
<style>
div {
width:400px;
height:80px;
background: -webkit-linear-gradient(red, blue);
background: -o-linear-gradient(red, blue);
background: -moz-linear-gradient(red, blue);
background: linear-gradient(red, blue);
}
</style>
</head>
<body>
<div></div>
</body>
</html>
```

这段代码根据不同内核的浏览器，设置了多种情况下线性渐变的效果。运行这段代码后，效果如图14.16所示。

对于从左到右的渐变，可以使用下面的代码，效果如图14.17所示。

```
background: -webkit-linear-gradient(left, red , blue);
background: -o-linear-gradient(right, red, blue);
background: -moz-linear-gradient(right, red, blue);
background: linear-gradient(to right, red , blue);
```

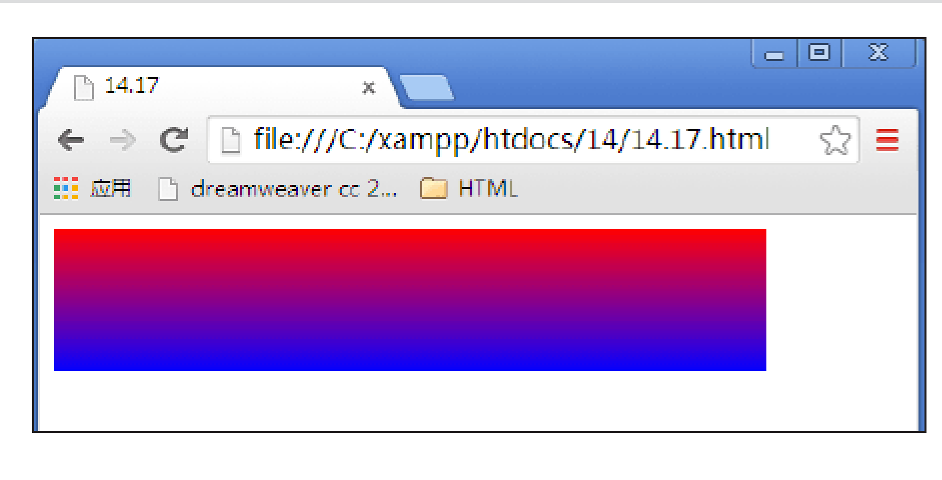


图14.16

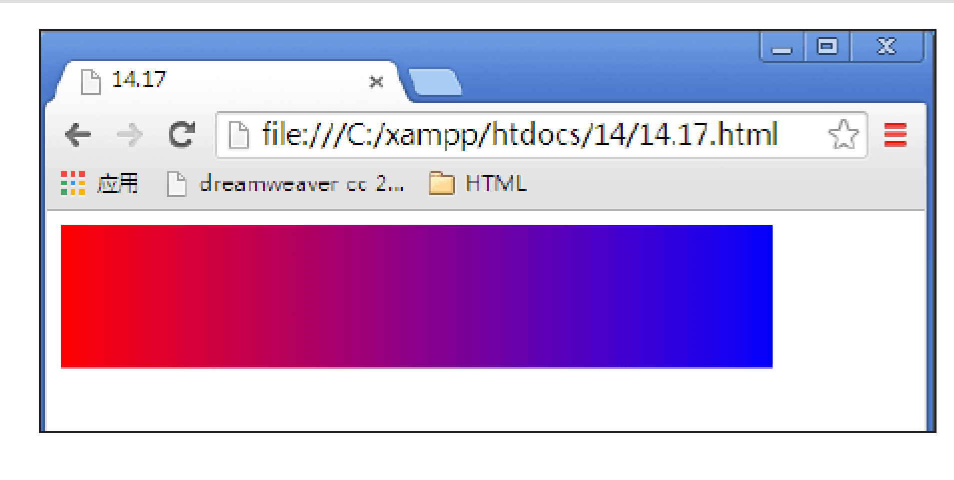


图14.17

在设置渐变起点的时候，可以设置一个对角点，这样就可以创建一个对角渐变的效果。相关代码如下，效果如图14.18所示。

```
background: -webkit-linear-gradient(left top, red , blue);
background: -o-linear-gradient(bottom right, red, blue);
background: -moz-linear-gradient(bottom right, red, blue);
background: linear-gradient(to bottom right, red , blue);
```

另外，还可以将渐变起点设置为一个角度，这个角度是指水平线和渐变线之间的角度，按逆时针方向计算。例如，0deg将创建一个从下到上的渐变，而90deg将创建一个从左到右的渐变。下面这段代码将创建一个30deg的线性渐变，效果如图14.19所示。

```
background: -webkit-linear-gradient(30deg, red, blue);
background: -o-linear-gradient(30deg, red, blue);
background: -moz-linear-gradient(30deg, red, blue);
background: linear-gradient(30deg, red, blue);
```

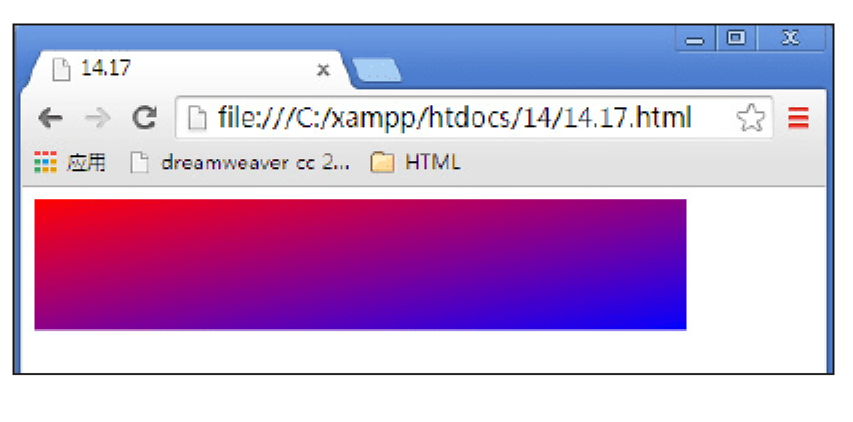


图14.18

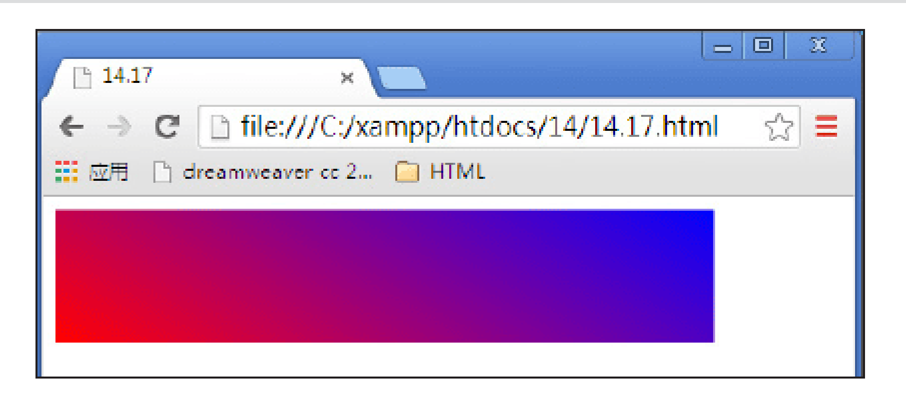


图14.19

14.6.2 径向渐变

径向渐变是从中心点向周围渐变。默认情况下，渐变的中心是center（表示在中心点），渐变的形状是ellipse（表示椭圆形），渐变的大小是farthest-corner（表示到最远的角落）。径向渐变的语法如下所示：

```
background: radial-gradient(center, shape size, start-color, ..., last-color);
```

如果仅仅指定渐变的颜色，那么颜色将会均匀的从中间向周围渐变。例如下面这段代码，运行后的效果如图14.20所示。

```
background: -webkit-radial-gradient(red, green, blue);
background: -o-radial-gradient(red, green, blue);
background: -moz-radial-gradient(red, green, blue);
background: radial-gradient(red, green, blue);
```

如果要改变各个颜色的分布情况，可以使用以下代码，效果如图14.21所示。

```
background: -webkit-radial-gradient(red 5%, green 15%, blue 60%);
background: -o-radial-gradient(red 5%, green 15%, blue 60%);
background: -moz-radial-gradient(red 5%, green 15%, blue 60%);
background: radial-gradient(red 5%, green 15%, blue 60%);
```

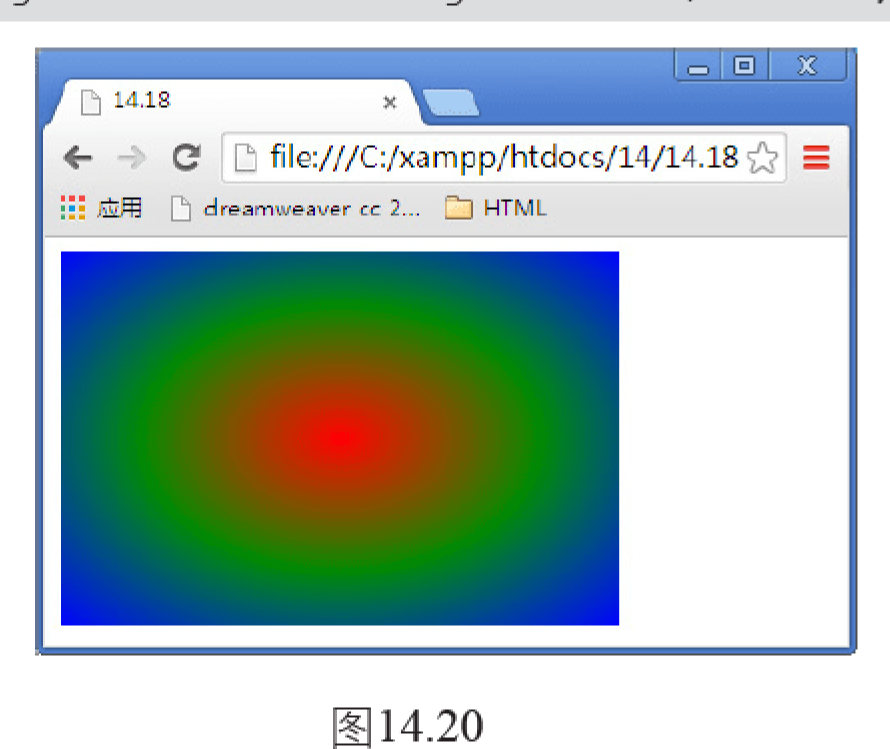


图14.20

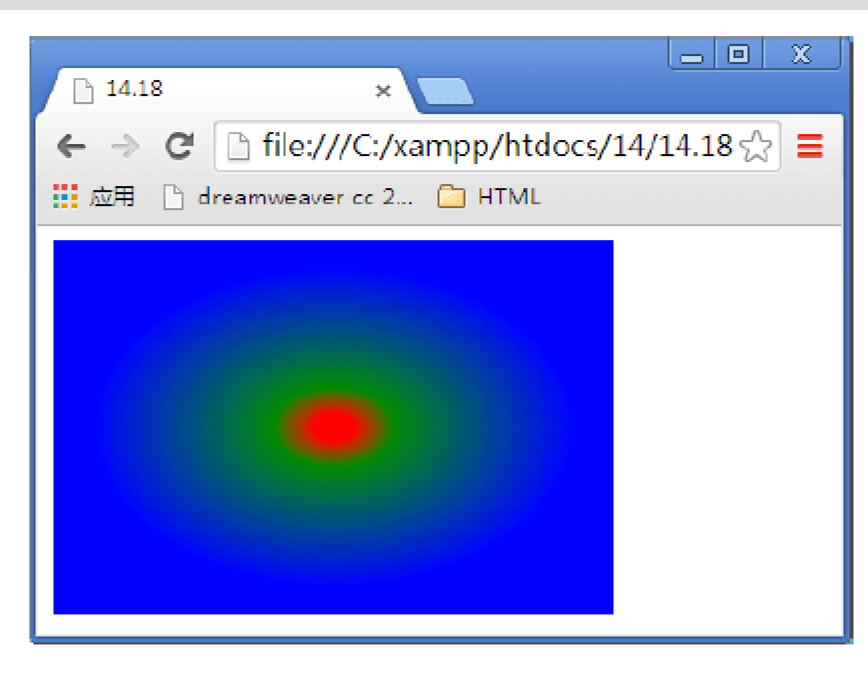


图14.21

径向渐变默认的渐变形状是圆形（circle），当然还可以通过以下设置将其改变为椭圆形

（ellipse），效果如图14.22所示。

```
    background: -webkit-radial-gradient(circle, red, yellow, green);
    background: -o-radial-gradient(circle, red, yellow, green);
    background: -moz-radial-gradient(circle, red, yellow, green);
    background: radial-gradient(circle, red, yellow, green);
```

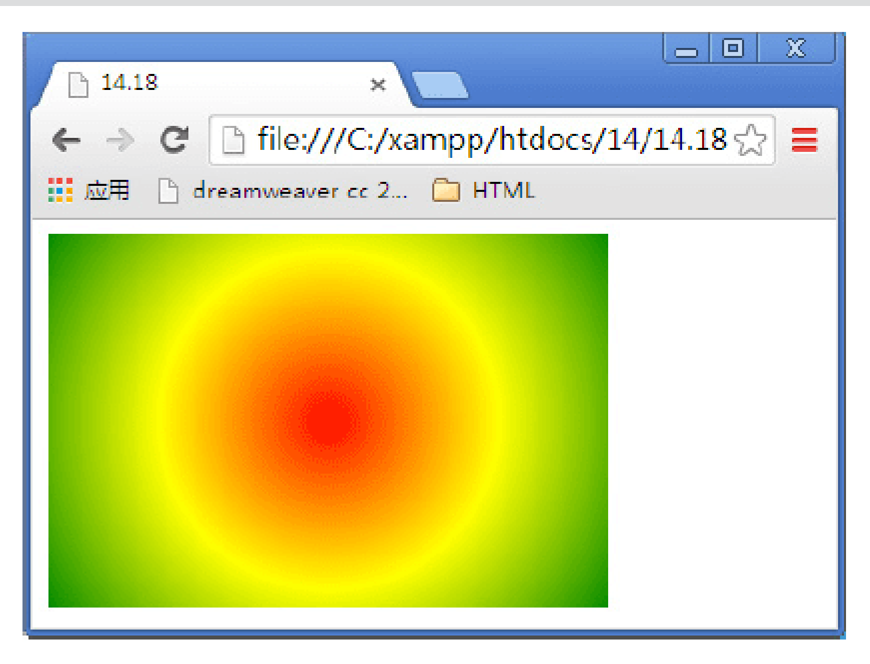


图14.22

径向渐变中的size参数用于设置渐变的大小，它有4种取值，closest-side的效果如图14.23所示，farthest-side的效果如图14.24所示，closest-corner的效果如图14.25所示，farthest-corner的效果如图14.26所示。

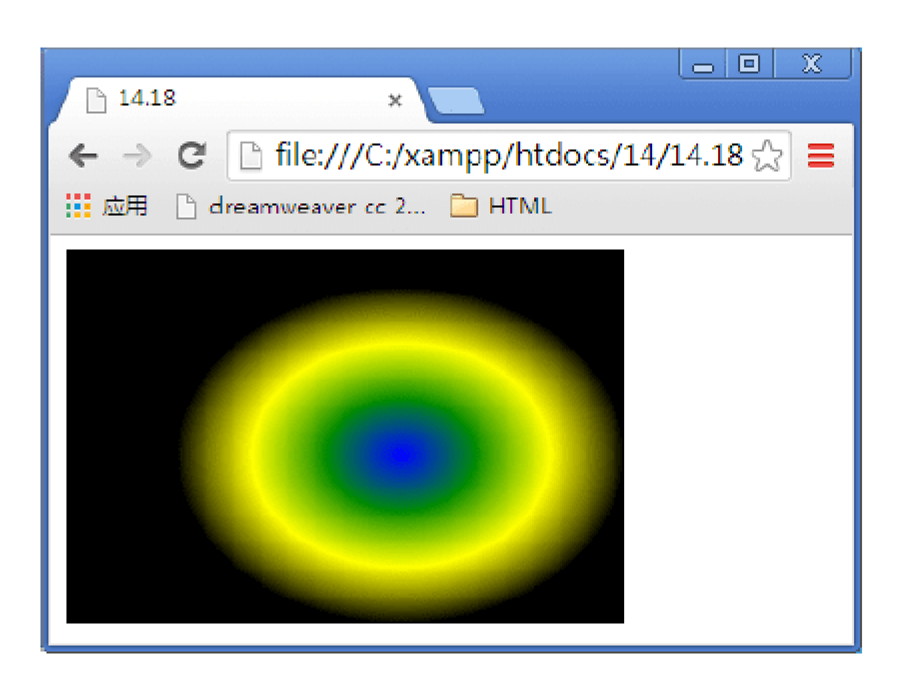


图14.23

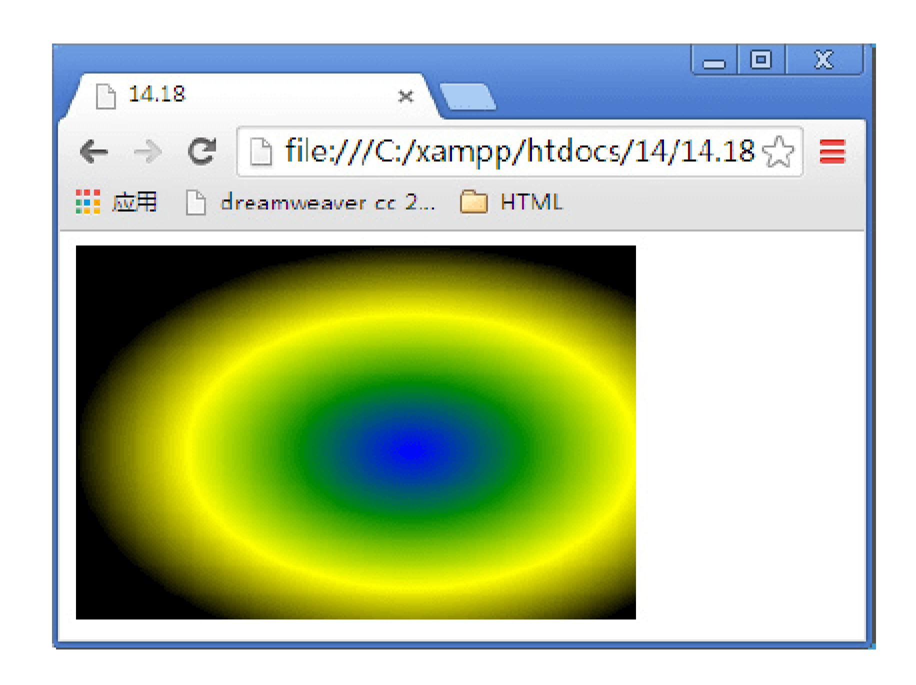


图14.24

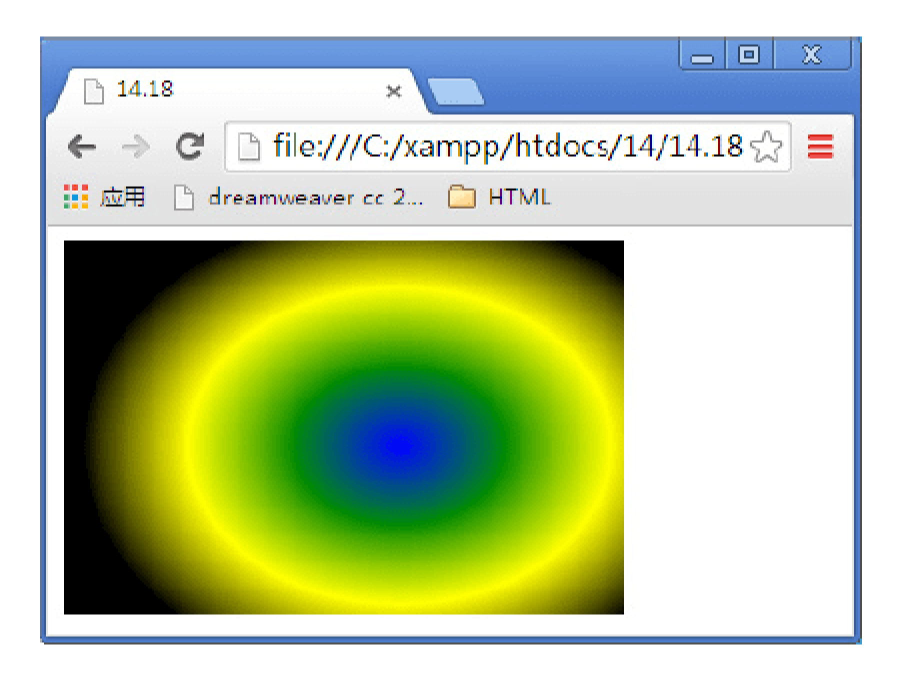


图14.25

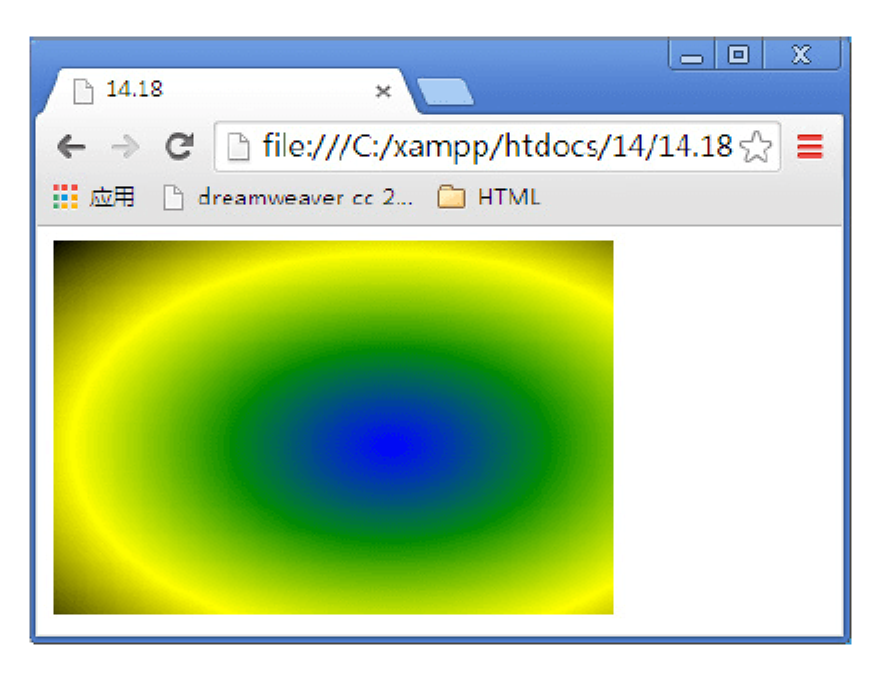


图14.26



14.7 CSS 3的阴影和反射效果

在CSS 3中可以轻松的创建阴影效果，创建阴影的时候需要4个参数，第1个参数表示X轴方向上的阴影，第2个单数表示Y轴方向上的阴影，第3个参数表示阴影的模糊半径，第4个参数表示阴影的颜色。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.19</title>
<style>
div{
width:100px;
height:80px;
background-color:#BBA6A7;
box-shadow:5px 5px 5px rgba(0,64,128,0.3);
}
h2{
text-shadow:5px 5px 5px rgba(64,64,64,0.5);
}
</style>
</head>
<body>
<div></div>
<h2>文字阴影</h2>
</body>
</html>
```

这段代码中分别为<div>元素和<h2>元素设置了阴影。不同的是，为<div>元素设置阴影的时候用的是box-shadow属性，而对<h2>元素设置阴影的时候用的是text-shadow属性。运行这段代码后，效果如图14.27所示。

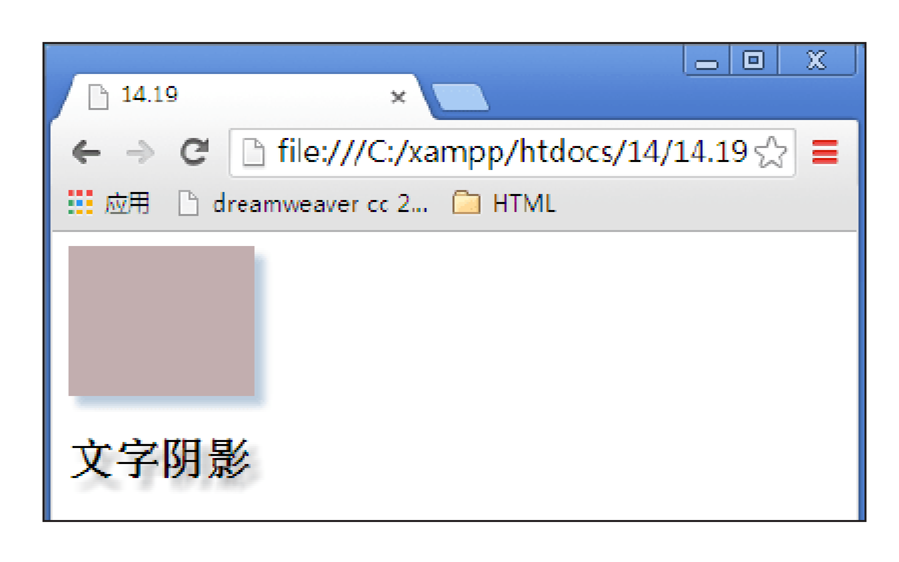


图14.27

使用CSS 3还可以创建反射效果。在创建反射时需要指定两个参数，第1个参数是反射的方向，第2个参数是反射的距离。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.20</title>
<style>
h1{
-webkit-box-reflect: below 5px
-webkit-gradient(linear, left top, left bottom, from(transparent),
to(rgba(255, 255, 255, 0.51)));
}
</style>
</head>
<body>
<h1>文字阴影</h1>
</body>
</html>
```

在这段代码中，我们不仅为文字设置了反射效果，还设置了透明效果。运行这段代码后，效果如图14.28所示。

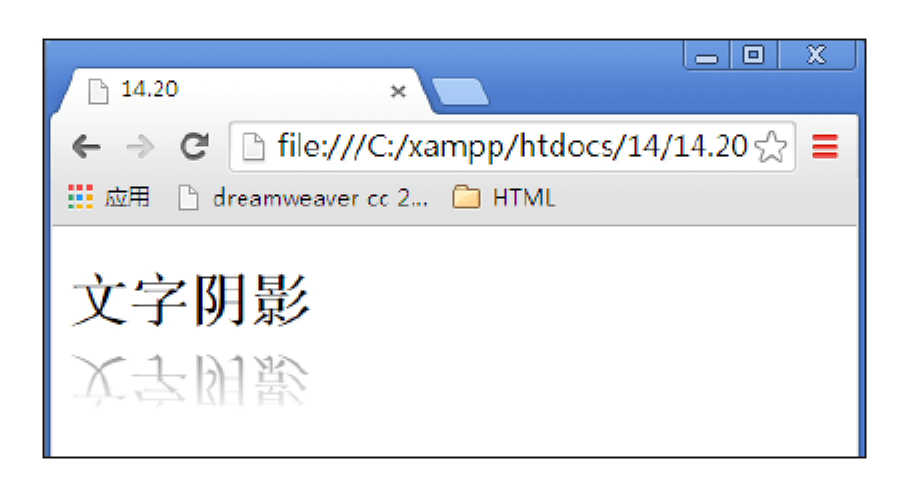


图14.28



14.8 CSS 3的背景效果

CSS 3中新增加了一些关于设置背景的属性，包括background-clip、background-origin和background-size，这些属性可以帮助用户创建多种样式的背景。另外，CSS 3还允许在一个元素上设置多种背景。

14.8.1 background-clip

该属性用于设置背景的区域，其默认值是border-box，表示背景绘制在边框方框内；另外还可以设置为padding-box，表示背景绘制在内边距的方框内；如果设置为content-box，则表示背景绘制在内容方框内。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.21</title>
<style>
h1{
padding:50px;
background-color:#B06667;
background-clip:content-box;
border:2px solid #C3181B;
}
</style>
</head>
<body>
<h1>这里是内容区域</h1>
</body>
</html>
```

在这段代码中，为<h1>元素设置了背景，同时设置背景显示区域为内容区域，效果如图14.29所示。

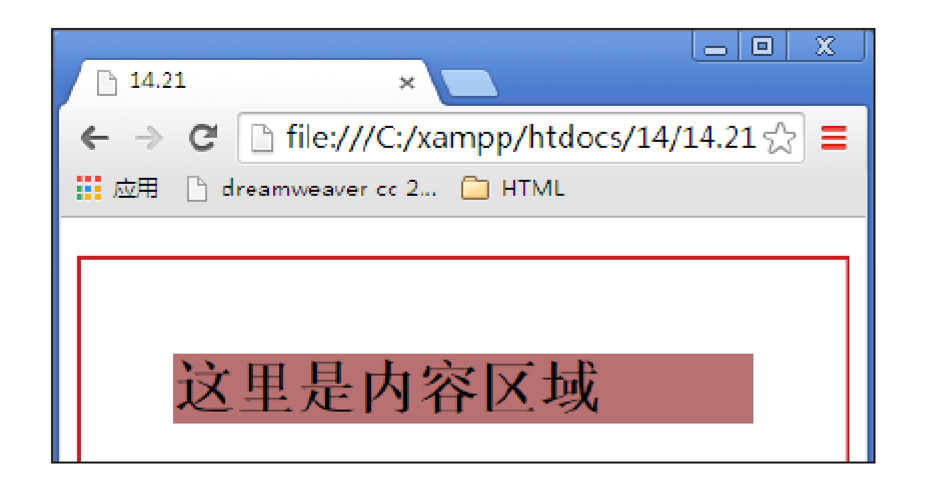


图14.29

14.8.2 background-origin

该属性用于确定背景的位置，它通常与background-position联合使用。如果设置为border-box则表示从border开始计算位置，如果设置为padding-box则表示从padding开始计算位置，如果设置为content-box则表示从content开始计算位置。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.22</title>
<style>
h1{
padding:50px;
background-image:url(img/img01.jpg);
background-repeat:no-repeat;
background-position:left;
```

```
background-origin:content-box;
border:2px solid #C3181B;
}
</style>
</head>
<body>
<h1>这里是内容区域</h1>
</body>
</html>
```

在这段代码中，background-position设置为left，background-origin设置为content-box，表示背景从内容区域的左边开始显示。运行这段代码后，效果如图14.30所示。

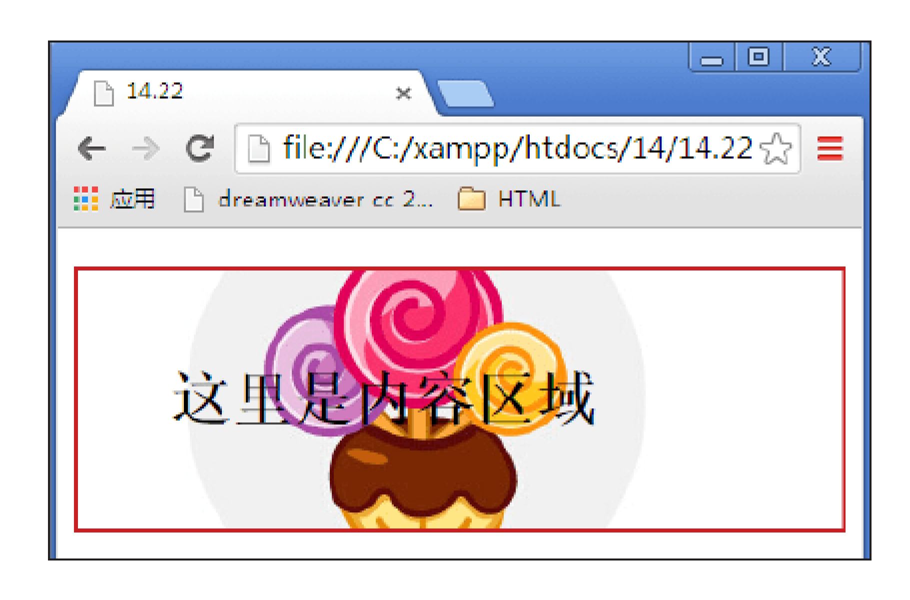


图14.30

14.8.3 background-size

该属性用于调整背景图片的大小。如果设置为contain，那么将缩小图片以适应元素，同时保持像素长宽比；如果设置为vover，将扩展图片以填补元素，同时保持像素长宽比；如果设置为两个具体长度，如background-size: 100px 100px，图片将被缩放到指定的尺寸；如果设置为百分比，如background-size: 50% 100%，图片将被缩放到指定的大小，百分比是相对于包含元素的尺寸。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.23</title>
<style>
h1{
padding:50px;
background-image:url(img/img01.jpg);
background-repeat:no-repeat;
background-size:contain;
border:2px solid #C3181B;
}
</style>
</head>
<body>
<h1>这里是内容区域</h1>
```

```
</body>
</html>
```

运行这段代码后，效果如图14.31所示。

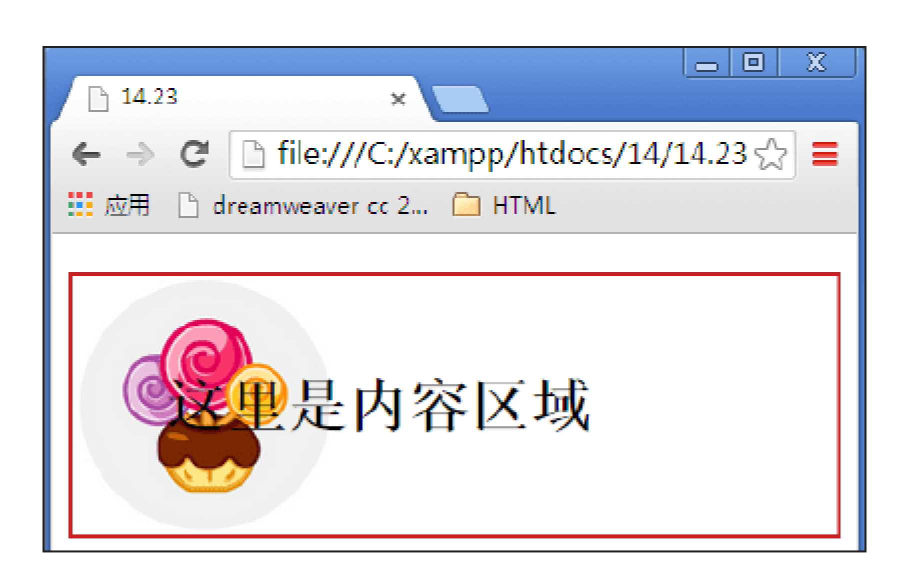


图14.31

14.8.4 设置多个背景

在CSS 3中允许为同一个元素设置多个不同的背景，例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>14.24</title>
<style>
h1{
padding:50px;
border:2px solid #C3181B;
background:url(img/img01.jpg) 10px center no-repeat, url(img/img02.jpg)
10px center repeat-x;
}
</style>
</head>
<body>
<h1>这里是内容区域</h1>
</body>
</html>
```

在这段代码中，使用background属性为<h1>元素设置了两个背景图片，效果如图14.32所示。

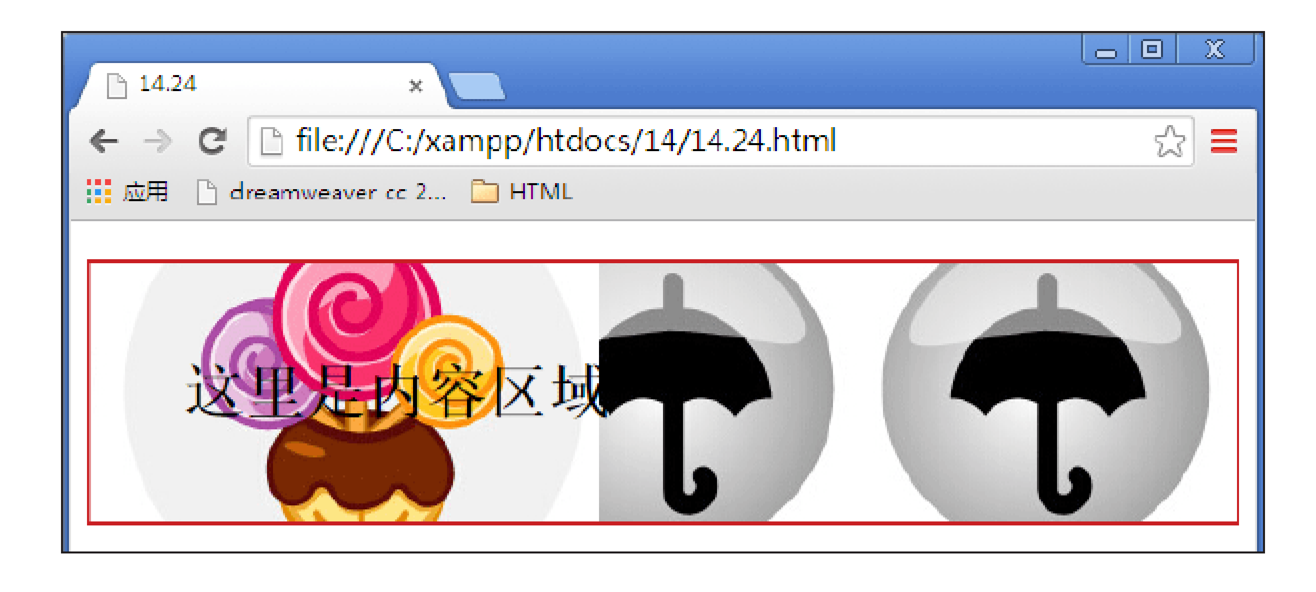


图14.32

第15章　响应式Web设计

当今智能设备充斥着我们生活的方方面面，各种移动设备正影响着我们的生活方式。智能手机、平板电脑和pc机都可以作为互联网的访问设备。不同设备的屏幕大小并不相同，各种设备对网站内容的兼容性也不同，而且并非所有人都需要在各自的设备上全屏浏览网站。为了解决这些问题，响应式Web设计应运而生。



15.1 什么是响应式Web设计

响应式Web设计(Responsive Web Design)的理念是页面的设计与开发应当根据设备环境（屏幕尺寸、屏幕定向、系统平台等）以及用户行为（改变窗口大小等）进行相应的响应和调整。响应式Web设计具体的实践方式由多方面组成，包括弹性网格和布局、图片、CSS media query的使用等。无论用户正在使用pc、平板电脑或者手机，是全屏显示还是非全屏的情况，屏幕是横向还是竖向，页面都应该能够自动切换分辨率、图片尺寸及相关脚本功能等。

对于响应式Web设计的页面而言，编程者只需要编写一次代码，创建一个在不同终端设备上有不同呈现形式的网页。下面我们来欣赏一些响应式网站设计的案例。



图15.1

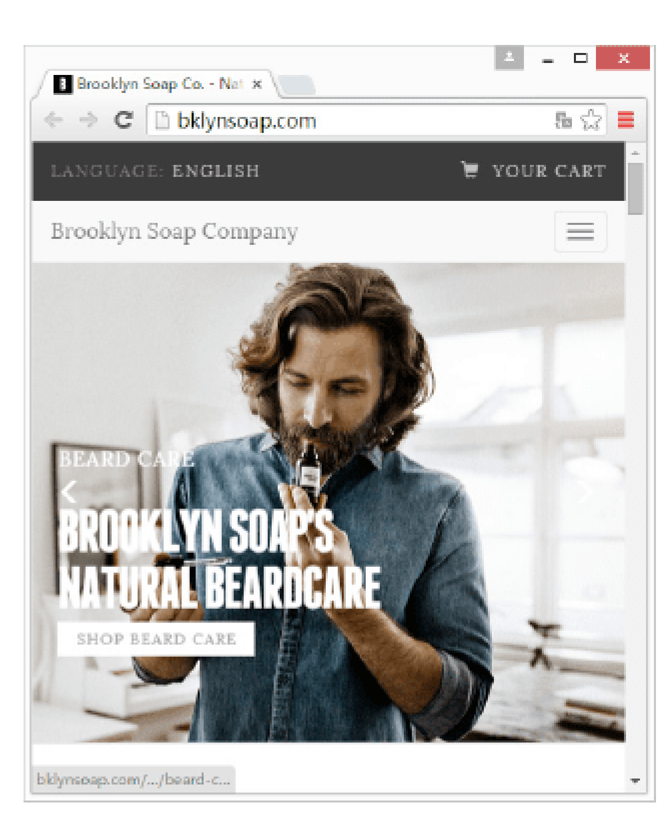


图15.2

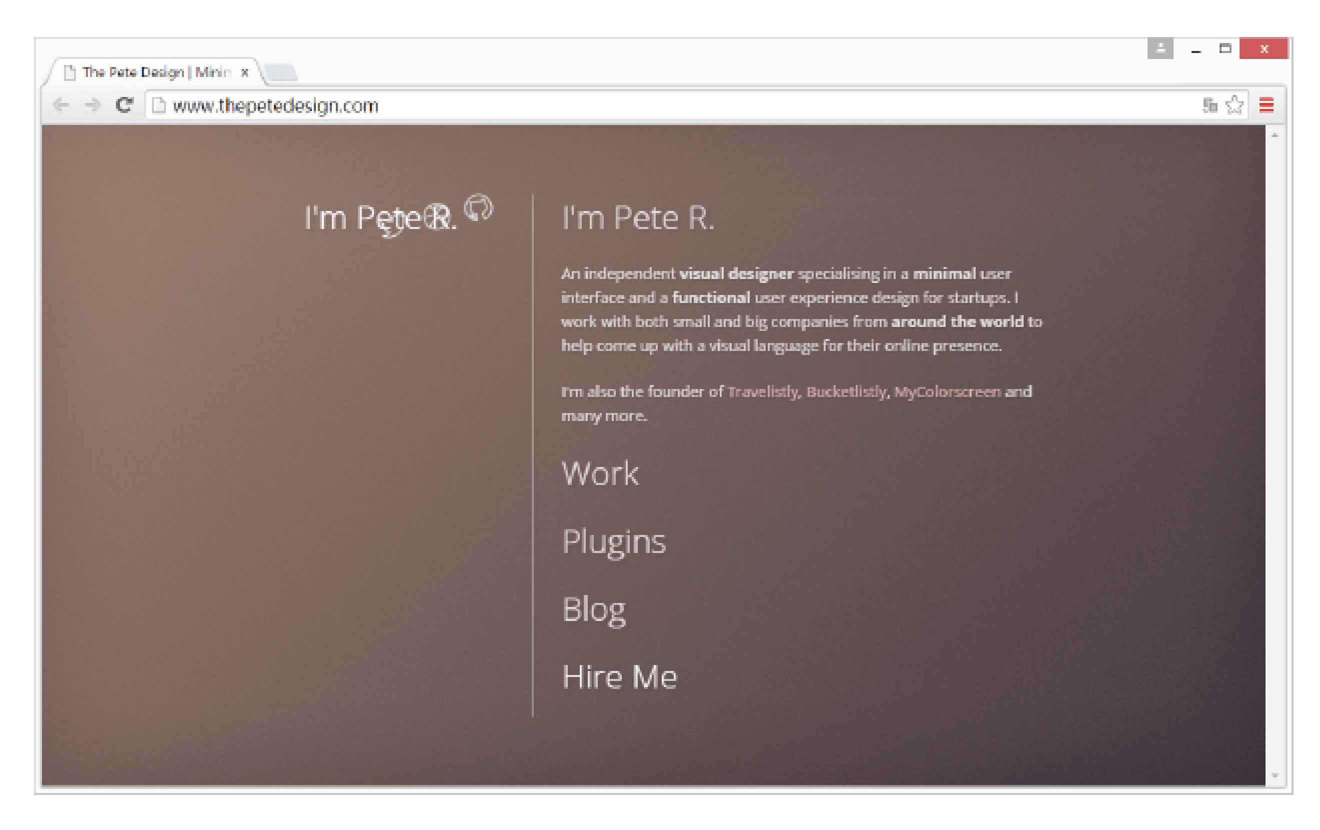


图15.3

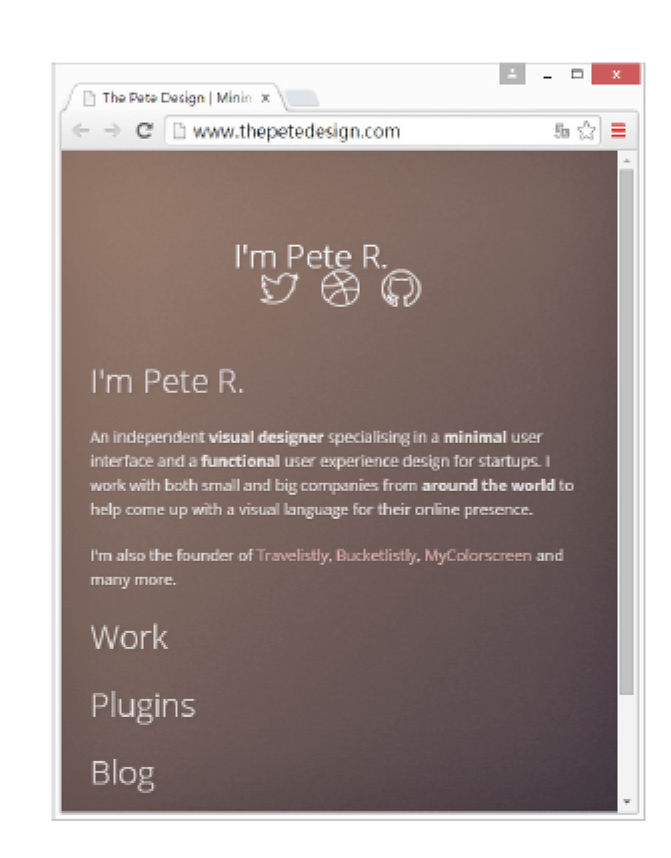


图15.4

图15.1页面是全屏显示的效果，我们可以看到网站的导航菜单有很多，背景图片只显示了部分图像，而图15.2是缩放浏览器窗口呈现的效果，导航菜单已经发生了变化，背景图片也显示了更多的范围，图像上面的文字也变得更大了。这些区别都是由于改变浏览器窗口而发生了变化，这就是响应式Web设计带给我们的惊喜。

我们再看图15.3所示页面的效果，页面的主体文字被一条垂直的直线分成两个部分，缩放浏览器窗口后原来左右排列的内容现在自动呈现为上下结构，效果如图15.4所示，这同样也是响应式Web设计带给我们的惊喜。



15.2 响应式Web设计的优势

由于技术上的进步，从移动设备上访问网站的数量越来越多，基于响应式Web设计的网站将是移动设备最理想的访问站点，因为它具有以下一些优势。

1. 成本优势

目前很多网站为了适应不同设备的浏览，会针对不同的设备开发多个版本，这样做的结果就是在开发、维护和运营上投入更多的成本。

基于响应式Web设计的网站，只有一个页面，只是针对不同的分辨率、不同的设备环境进行了一些不同的设计，所以在开发、维护和运营上，能有效节约成本。

2. 兼容性优势

移动设备新的尺寸层出不穷，定制的版本通常只适用于某些规格的设备。如果新的设备分辨率变化较大，则往往不能兼容，而开发新的版本需要时间，这段时间内的访问就是个问题，但是响应式Web设计可以提前预防这个问题。

3. 操作灵活

响应式设计是针对页面的，可以只对必要的页面进行改动，其他页面不受影响。

响应式Web设计虽然灵活性强，可以适应不同分辨率的设备，但是它的缺点也是显而易见的。为了兼容各种设备，就需要不断扩展代码，最终导致代码臃肿，而且页面加载的时间也会比较长。所以，针对项目的实际情况，有选择的采用响应式Web设计是比较推荐的选择。



15.3 制作响应式Web设计的方法

在制作响应式Web设计的时候，我们应该遵循以下设计流程。

（1）确定需要兼容的设备类型、屏幕尺寸：调查研究用户使用各种设备的分布情况，确定需要兼容哪些设备，哪些尺寸。

目前市面上的移动设备和平板电脑的种类都很多，而且大多具有不同的屏幕尺寸，pc机显示器的尺寸也分了很多种。在设计响应式Web设计的时候，要充分考虑到手势功能、屏幕尺寸、横向和竖向、普通电脑屏幕和宽屏的设计。

对于现有的网站，并非需要将网站上所有的页面都设计为响应式Web页面，仅需要将移动设备可能访问比较频繁的页面设计为响应式Web页面即可。

（2）制作线框原型：针对确定下来的几个尺寸分别制作不同的线框原型。需要考虑清楚在不同尺寸下，页面的布局如何变化，内容尺寸如何缩放，功能、内容如何删减，甚至针对特殊的环境作特殊化的设计等。这个过程需要设计师和前端开发人员保持密切沟通。

（3）测试线框原型：将图片导入到相应的设备进行一些简单的测试，可帮助我们尽早发现在可访问性、可读性等方面存在的问题。

（4）视觉设计：移动设备的屏幕像素密度与传统电脑屏幕不一样，在设计的时候需要保证内容文字的可读性、控件可点击区域的面积等。

（5）前端实现：与传统的web开发相比，响应式设计的页面由于页面布局、内容尺寸发生了变化，所以最终的产出更有可能与设计稿出入较大，需要前端开发人员和设计师多沟通。



15.4 视口和屏幕尺寸

视口是指浏览器窗口内的内容区域。例如可以按下F12功能键，打开谷歌浏览器的开发者工具界面，然后点击如图15.5所示图形左下角的手机图标按钮，打开视口调试窗口。这个窗口内显示的区域就是视口，可以使用鼠标拉伸视口的边缘改变视口的大小，或者直接在视口左上角的设备（Device）下拉框中选择指定的设备，以便确定视口的大小。

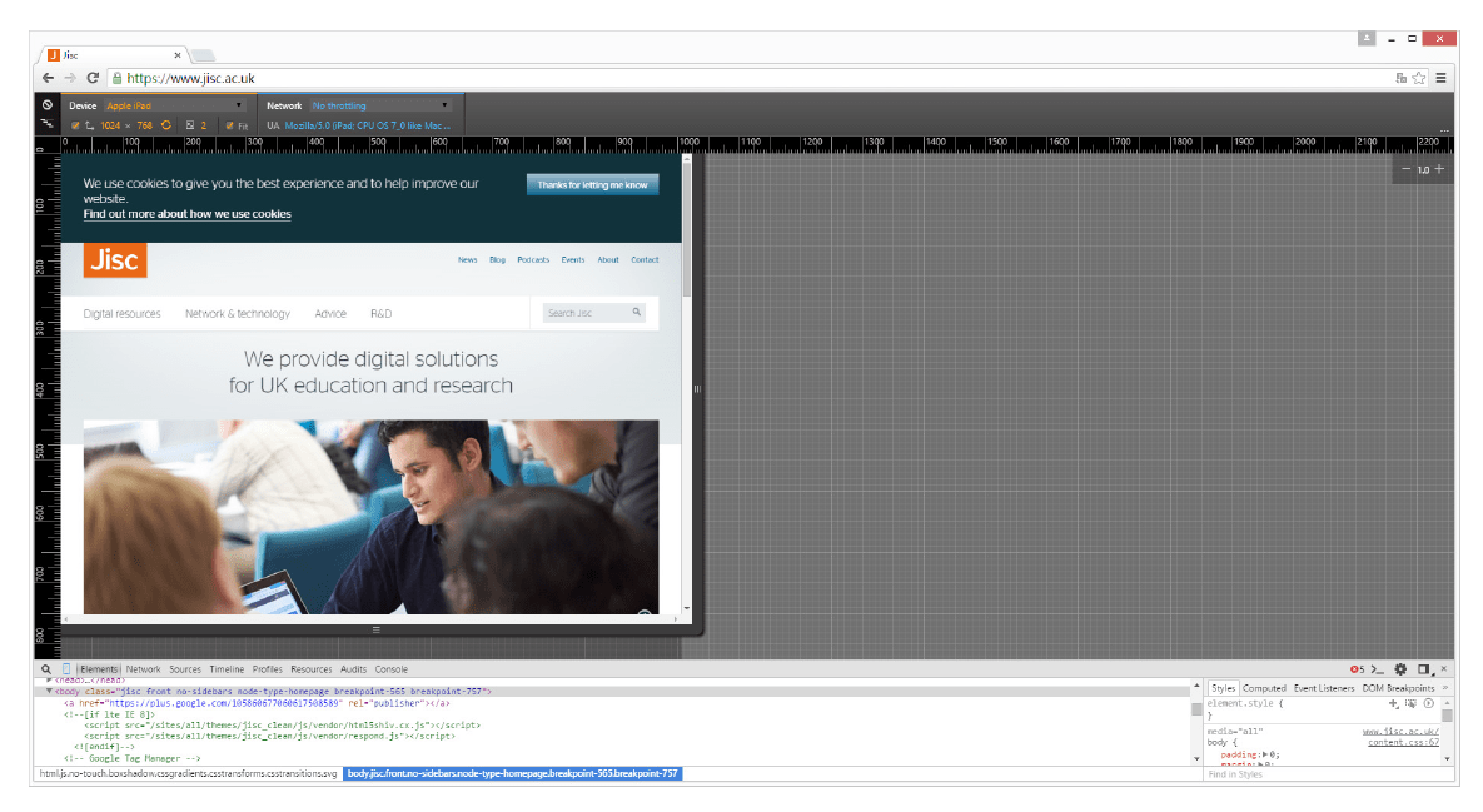


图15.5

屏幕尺寸指的是设备的物理显示区域，也就是用户屏幕的整体大小，它与视口是两个不同的概念。



15.5 媒体查询

HTML中的<link>标签有一个media的属性，该属性为样式表指定设备的类型，例如以下两段代码：

```
<link rel="stylesheet" type="text/css" media="screen" href="screen.css" />
<link rel="stylesheet" type="text/css" media="print" href="print.css" />
```

这两行代码同时出现在一个HTML页面中，为不同的设备指定加载不同的样式文件。第一行代码表示页面显示在屏幕上时，使用screen.css这个样式文件；而打印这个页面时，使用print.css这个样式文件。

在CSS 3中，媒体查询（Media Query）可以设置不同类型的媒体条件，并根据不同的条件指定不同的样式文件。可以将媒体查询语句看成是一个表达式，当表达式满足不同条件时，应用不同的样式。表15.1列出了媒体查询中不同条件的媒体功能。

表15.1

功能	值	最大/最小值	描述
width	长度	是	显示区域的宽度
height	长度	是	显示区域的高度
device-width	长度	是	设备的宽度
device-height	长度	是	设备的高度
orientation	portrait或landscape	否	设备的方向
aspect-ratio	宽度比（宽/高）	是	设备的宽高比，使用由1个斜杠分开的两个整数表示（比如16/9）
device-aspect-ratio	高宽比（宽/高）	是	设备宽度与设备高度的比率
color	整数	是	每种颜色成分的位数（如果不是颜色，该值为0）
color-index	整数		每种颜色成分的位数（如果不是颜色，该值为0）
monochrome	整数		输出设备的颜色并查找表中的项数
resolution	分辨率		输出设备的像素密度，表示为整数后跟dpi（每英寸点数）或dpcm（每厘米点数）
scan	Progressive或interlace	否	TV设备使用的扫描过程
grid	0或1	否	如果设置为1，设备基于网格，比如电传类型的终端或仅有一种固定字体的电话显示设备（所有其他设备均为0）

例如，我们需要为一个纵向显示的屏幕设置样式，可以使用以下的代码：

```
<link rel="stylesheet" type="text/css" media="screen and (orientation:portrait)" href="screen_portrait.css" />
```

如果要为一个横向显示的屏幕设置样式，可以使用以下的代码：

```
<link rel="stylesheet" type="text/css" media="screen and (orientation: landscape)" href="screen_ landscape.css" />
```

或者在媒体查询纵向样式的前面添加not关键字，例如下面的代码：

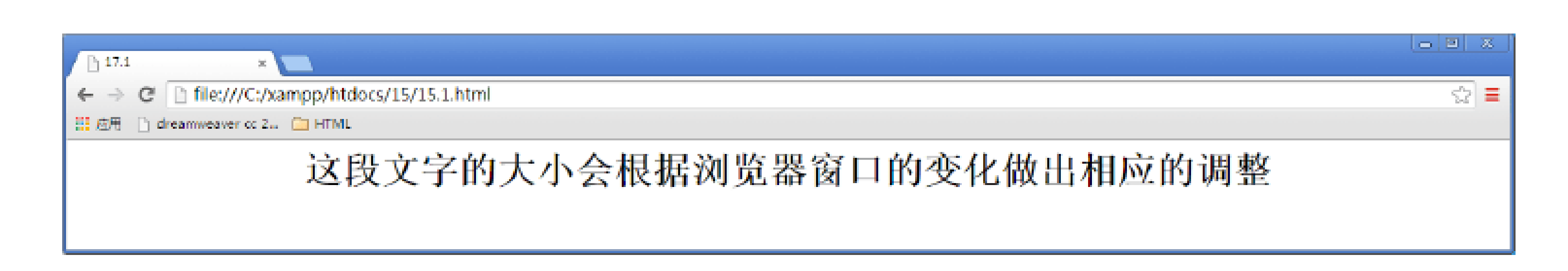
```
<link rel="stylesheet" type="text/css" media="not screen and (orientation:portrait)" href="screen_ landscape.css" />
```

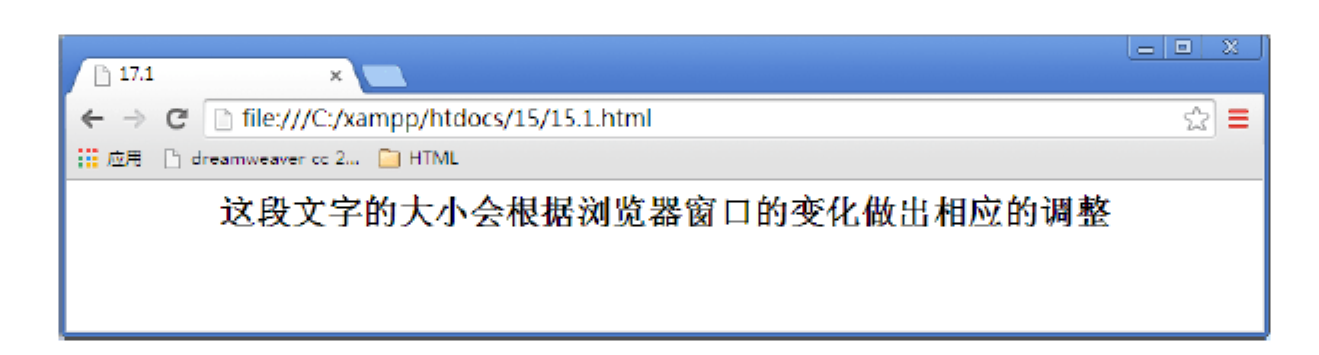
在一个媒体查询条件中，可以设置多个查询条件，多个查询条件之间用and关键字连接。例如查询视口宽度大于550px的纵向屏幕，可以设置以下媒体查询条件：

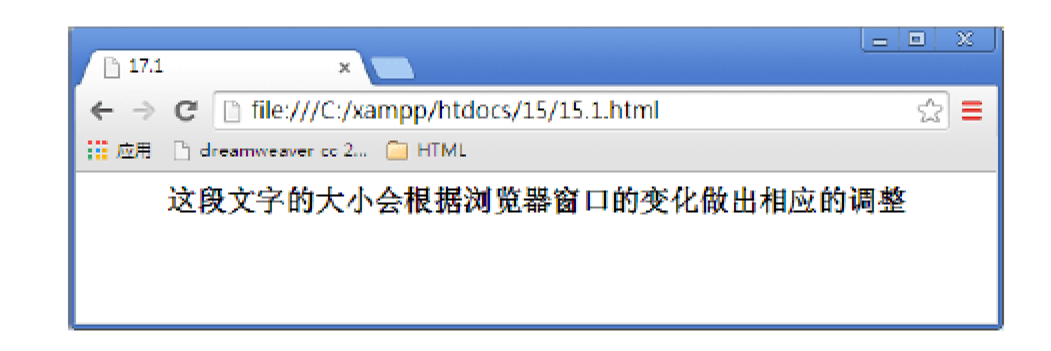
```
<link rel="stylesheet" type="text/css" media="not screen and (orientation:portrait) and (min-width:550px) " href="screen_portrait_width550.css" />
```

以上介绍的是使用多个样式表文件时媒体查询的使用方法，但是这种方法会导致开发人员分别存储多套样式表文件，当需要修改样式的时候，会显得非常麻烦，而且页面在加载多个样式表的时候，会因为重复访问而降低加载速度。所以，笔者推荐的方法是在统一一个样式表中，将多套媒体查询集中写在一起，这样当需要修改的时候，可以非常方便地定位目标，而且不容易遗漏。例如在下面的代码中，通过媒体查询设置浏览器在不同尺寸视口时，页面中字体的大小变化效果。运行该页面，调整浏览器窗口的宽度，可以看到字体的变化效果，如图15.6所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>15.1</title>
<style>
body {
font-size:36px; text-align:center; font-weight:bold;
}
@media screen and (max-width: 960px) {
body {
font-size:24px; text-align:center; font-weight:bold;
}
}
@media screen and (max-width: 768px) {
body {
font-size:20px; text-align:center; font-weight:bold;
}
}
@media screen and (max-width: 550px) {
body {
font-size:16px; text-align:center; font-weight:bold;
}
}
@media screen and (max-width: 320px) {
body {
font-size:9px; text-align:center; font-weight:bold;
}
}
</style>
</head>
<body>
<div>这段文字的大小会根据浏览器窗口的变化做出相应的调整</div>
</body>
</html>
```







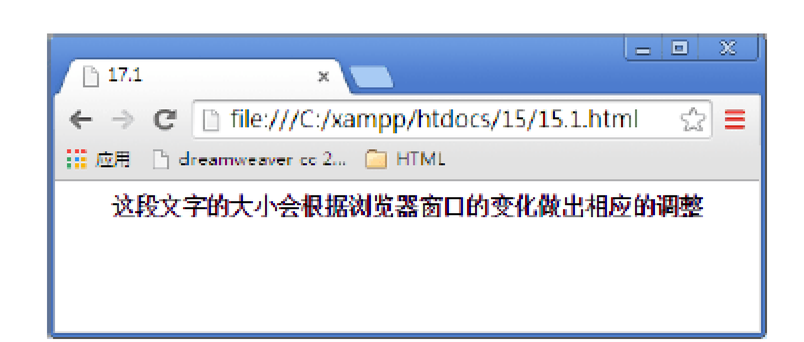


图15.6



15.6 响应式图片

在编写CSS代码的时候，如果要指定一个图像的尺寸，可以使用像素（px）为单位，也可以使用百分比。如果使用像素为单位，那么无论怎么调整视口，图像的大小都不会发生变化；如果使用百分比来设置图像，当调整视口的大小时，图像的尺寸也会随着变化。但是，如果视口尺寸大于图像的原始尺寸，图像就会变形。为了让图像保持最大原始尺寸，可以通过以下代码设置。

```
img{max-width:300px;}
```

图像实现了弹性拉伸，但这并不是响应式图片的真正含义。响应式图片是指根据设备的屏幕尺寸提供不同的图片，当切换设备的时候，能够正确显示对应尺寸的图片。

第16章 PHP动态网站开发

动态网站的网站内容可以根据不同的情况进行变更，通常情况下是指客户端与服务端可以进行交互的一种网站，客户端是指Web页面，而服务端多为数据库架构。所以动态网站不仅包括客户端的网页设计，而且还需要通过数据库和编程来对网站内容进行更新。



16.1 动态网站开发基础

目前几乎所有的网站都是动态网站。因为有了动态网站，才使用户与Web页面有了更多的交互，才会有更多更新颖的网站涌现出来，从而不断推动互联网技术向新的方向发展。

16.1.1 功能特点

动态网站之所以成为目前互联网发展的方向，这是因为它具有以下几个特点：

（1）动态网站可以实现交互功能，如用户注册、信息发布、产品展示、信息管理等。

（2）动态网页需要客户端浏览器发出请求后才能实现交互。

（3）动态网页的后缀名一般是jsp、asp、php等。并非所有后缀名为HTML的网页都是静态网页，动态网页也可以使用URL静态化技术，使网页后缀显示为HTML。所以后缀名并非是判断网站是静态或动态的唯一标准。

（4）动态网站因为需要与数据库交互，所以访问速度受到一定影响。

（5）搜索引擎对动态网页的收录相对静态网页要弱一些。

16.1.1 开发语言

目前用于动态网站开发的语言主要有4种，ASP、ASP.NET、PHP和JSP。关于这4种语言，下面做一个简单的介绍。

（1）ASP是Active Server Pages（动态服务器网页）的简称，它是微软开发的一种类似超文本标识语言(HTML)、脚本(Script)与公用网关接口（CGI）的结合体，它没有提供自己专门的编程语言，而是允许用户使用许多已有的脚本语言编写ASP的应用程序。ASP的程序编制

比HTML更方便且更有灵活性。ASP是在Web服务器端运行，运行后再将运行结果以HTML格式传送至客户端的浏览器。因此，与一般的脚本语言相比，ASP要安全得多。

ASP的最大好处是可以包含HTML标签，也可以直接存取数据库及使用无限扩充的ActiveX控件，因此在程序编制上要比HTML方便而且更富有灵活性。通过使用ASP的组件和对象技术，用户可以直接使用ActiveX控件，调用对象方法和属性，以简单的方式实现强大的交互功能。

但ASP技术也非完美无缺，由于它基本上局限于微软的操作系统平台之上，主要工作环境是微软的IIS应用程序结构，又因ActiveX对象具有平台特性，所以ASP技术实现在跨平台Web服务器上工作不是很容易。

（2）ASP .NET的前身ASP技术，是在IIS 2.0上首次推出的，当时与 ADO 1.0一起推出，在IIS 3.0上成为服务器端应用程序的热门开发工具，微软还特别为它量身打造了VisualInter Dev开发工具。在1994年到2000年之间，ASP技术已经成为微软推展Windows NT 4.0平台的关键技术之一，数以万计的ASP网站也是从这个时候开始如雨后春笋般地出现在网络上。ASP的简单并且高度可定制化的能力，也是它能迅速崛起的原因之一。不过ASP的缺点也逐渐浮现出来。面向过程型的程序开发方法，让维护的难度提高很多，尤其是大型的ASP应用程序。 解释型的VBScript或JScript语言，让性能无法完全发挥； 扩展性由于其基础架构的不足而受限，虽然有COM元件可用，但开发一些特殊功能（如文件上传）时，没有来自内置的支持，需要寻求第三方控件商的控件。

（3）PHP也称为Hypertext Preprocessor（超文本预处理器），它是当今Internet上最为火热的脚本语言，其语法借鉴了C、Java、PERL等语言，只需要很少的编程知识你就能使用PHP建立一个真正交互的Web站点。

PHP与HTML语言具有非常好的兼容性，使用者可以直接在脚本代码中加入HTML标签，或者在HTML标签中加入脚本代码从而更好地实现页面控制。PHP提供了标准的数据库接口，数据库连接方便、兼容性强、扩展性强，可以进行面向对象编程。

（4）JSP也称Java Server Pages（java服务器页面），它是由Sun Microsystems公司于1999年6月推出的新技术，是基于Java Servlet以及整个Java体系的Web开发技术。

JSP和ASP在技术方面有许多相似之处，不过两者来源于不同的技术规范组织。ASP一般只应用于Windows NT/2000平台，而JSP可以在85%以上的服务器上运行，而且基于JSP技术的应用程序比基于ASP的应用程序易于维护和管理，所以被许多人认为是未来最有发展前途的动态网站技术。



16.2 PHP语言入门

PHP脚本语言的语法结构与C语言和Perl语言的语法风格非常相似。用户在使用变量前不

需要对变量进行声明。使用PHP创建数组的过程也非常简单。PHP还具有基本的面向对象组件功能，可以极大地方便用户有效组织和封装自己编写的代码。

16.2.1 PHP代码书写

PHP代码标注的书写格式以<?php开始，以?>结束，例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.1</title>
</head>
<body>
<?php
echo "这是第一个PHP案例！";
?>
</body>
</html>
```

在这段代码中，我们可以看到在HTML页面的<body>元素中有一段php代码，其中echo在php中表示输出的意思。运行这段代码后，效果如图16.1所示。

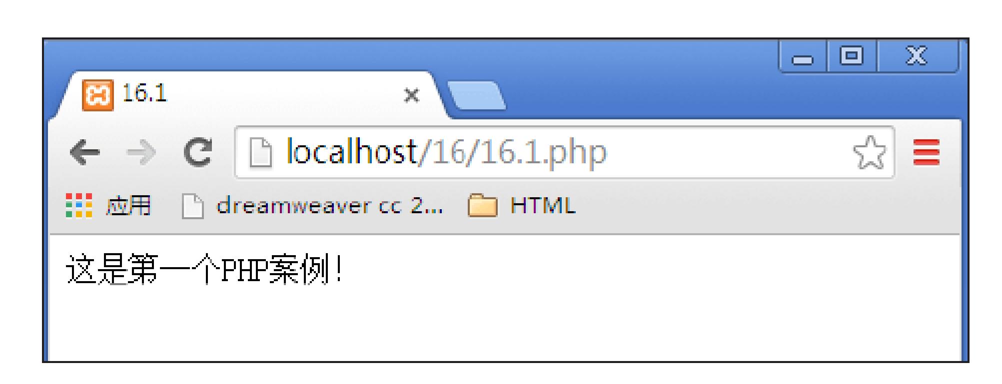


图16.1

除此之外，PHP代码还有一种简写方法。例如，下面这段代码与上面案例中代码的效果完全相同，不同的是这段代码中省略了php三个字母，这称之为PHP的简写风格。

```
<?
echo "这是第一个PHP案例！";
?>
```

另外，PHP还可以像JavaScript语言那样，使用script进行声明，这种写法如下所示：

```
<script language="php">
echo "这是第一个PHP案例！";
</script>
```

虽然PHP代码有多种写法，但是为了达到更好的兼容性，我们推荐使用标准风格，而不推荐使用剪短风格。

16.2.2 PHP代码注释

每一种编程语言都有自己的注释方法。在PHP中，使用“//”来编写单行注释，使用

“/*…*/”来编写多行注释，也可以使用“#”来注释。需要注意的是，注释的内容应该写在代码的上方或右方，不应该与代码混写，而且注释的内容不会显示在页面上。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.2</title>
</head>
<body>
<?php
//这里是PHP的单行注释内容
/*
这里是PHP的多行注释内容，注释的内容
都不会显示在页面上。
*/
echo "注释的内容不会显示出来。";
?>
</body>
</html>
```

运行这段代码后，页面上只会显示echo输出的内容，其他注释的内容不会显示在页面中，效果如图16.2所示。

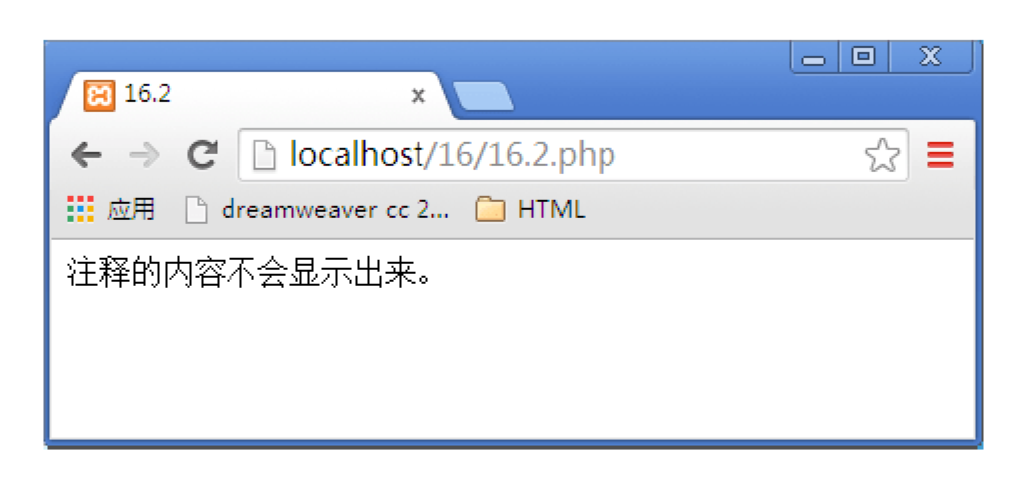


图16.2

16.2.3 PHP输出函数

PHP中总共有4类输出函数，分别是echo()函数、print()函数、printf()函数和sprintf()函数，这些函数各自都有不同的用法。

1. echo()函数

echo()函数用于输出一个或多个字符串，无返回值。可以用圆括号，也可以不用。通常情况下，我们直接省略圆括号，例如下面这段代码：

```
<?php
echo("带圆括号输出内容。");
echo "不带圆括号输出内容。" ;
?>
```

2. print()函数

print()函数用于输出一个或多个字符串，执行成功时返回1，执行失败时返回false。与echo函数一样，print()函数也可以省略圆括号。例如下面这段代码：

```
<?php
print("带圆括号输出内容。<br>");
print "不带圆括号输出内容。<br>" ;
echo print "带返回值输出。" ;
?>
```

运行这段代码后，效果如图16.3所示。在第3行输出的内容后面有一个1，这是因为第3行使用了两个输出函数，print函数先输出了内容，返回了一个1，然后echo函数又将全部内容输出。

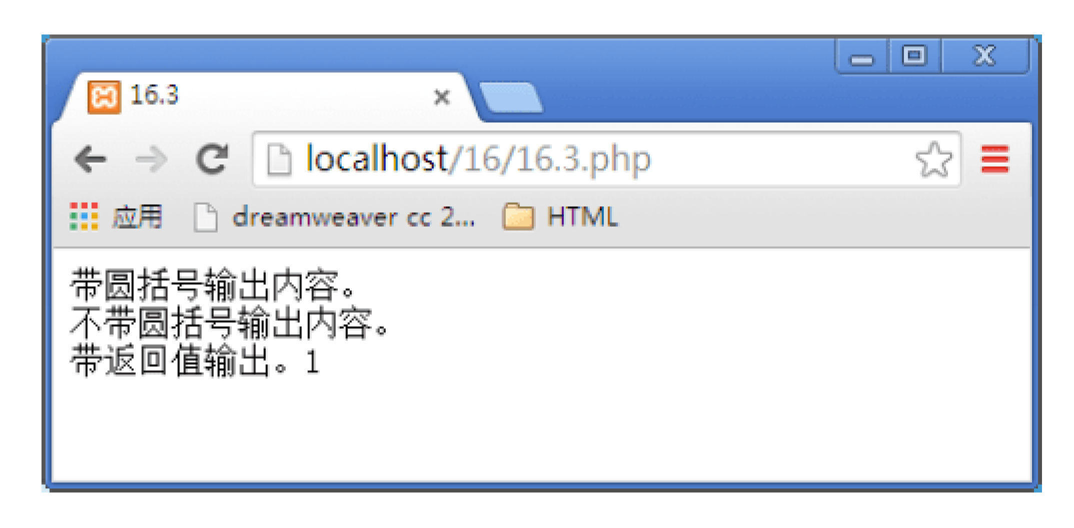


图16.3

3. printf()函数

printf()函数用于输出格式化的字符串。其中格式化字符串包括两部分内容，一部分是正常字符，这些字符将按原样输出；另一部分是格式化规定的字符，这些字符以百分号（%）开始，后面跟一个或几个规定的字符，用来确定输出内容格式。这些规定的字符如表16.1所示。

表16.1

转换符号	说明
%b	整数转二进制
%c	整数转ASCII码
%d	整数转有符号十进制
%f	双精度转浮点
%o	整数转八进制
%s	整数转字符串
%u	整数转无符号十进制
%x	整数转十六进制（小写）
%X	整数转十六进制（大写）

在格式化输出时，需要指定转换符号和对应的参数，而且每一个转换符号和对应的参数都必须一一对应，多个参数之间用逗号隔开，如果对应出错，会出现意想不到的错误。例如下面这段代码：

```
<!doctype html>
```

```
<html>
<head>
<meta charset="utf-8">
<title>16.5</title>
</head>
<body>
<?php
$name="小明";
$age=12;
printf("%s过完年就%u岁了。",$name,$age);
?>
</body>
</html>
```

在这段代码中使用了两个转换符号，%s对应变量$name，%u对应变量$age，他们是一一对应的关系。运行这段代码后，效果如图16.4所示。

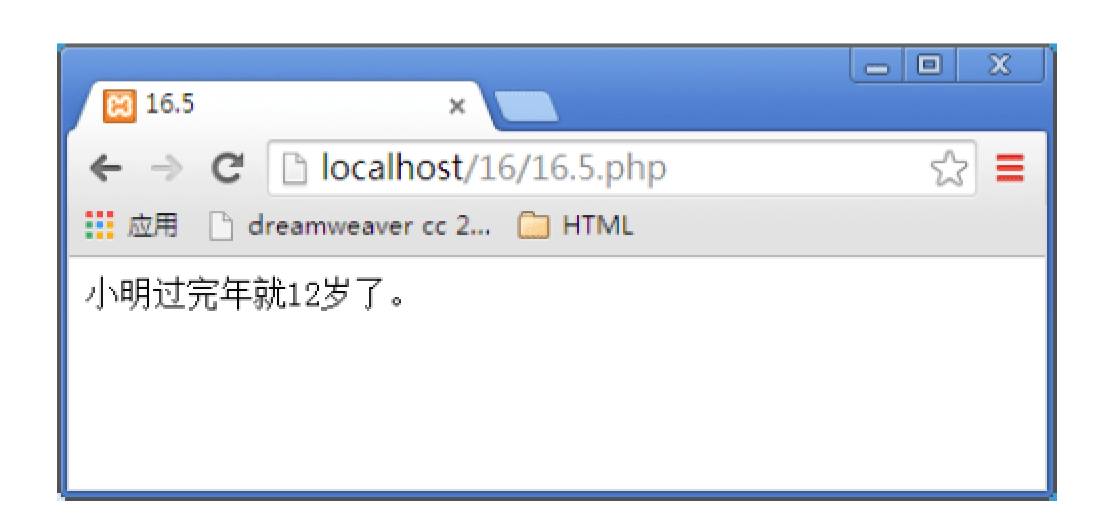


图16.4

printf函数有一个返回值，表示输出字符串的函数。要想得到这个返回值，需要使用each函数将其输出，相关代码如下所示：

```
echo printf("%s过完年就%u岁了。",$name,$age);
```

刷新页面后，效果如图16.5所示，后面多出来的29就是前面字符串的长度。需要说明的是，一般情况下我们认为一个汉字是两个字节长度，但是在编码为utf-8时，一个汉字的长度可能是2~3个字节，因为它是边长编码。

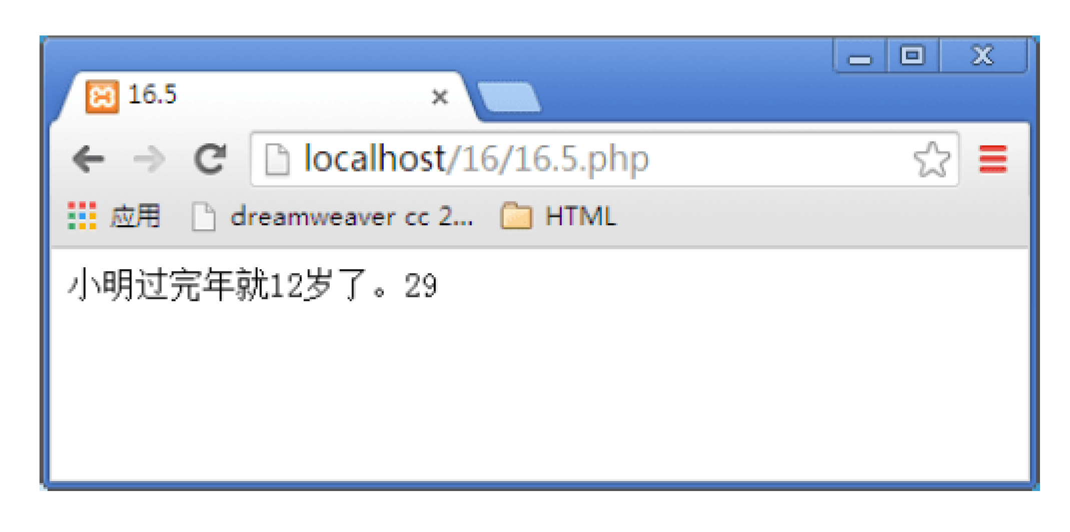


图16.5

4. sprintf()函数

sprintf()函数与printf()函数类似，printf()函数的返回值是字符串的长度，而sprintf()函数的返回值是字符串本身。因此，sprintf()函数必须通过echo才能输出。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.6</title>
</head>
<body>
<?php
$name="小明";
$age=12;
echo sprintf("%s过完年就%u岁了。",$name,$age);
?>
</body>
</html>
```

如果这里省略了echo函数，那么页面上将无法输出任何内容。运行这段代码后，效果如图16.4所示。sprintf()函数常用于数值转换，例如将十进制数转换为二进制数，可以执行以下代码：

```
<?php
$number=23;
echo sprintf("%b",$number);
?>
```

效果如图16.6所示。

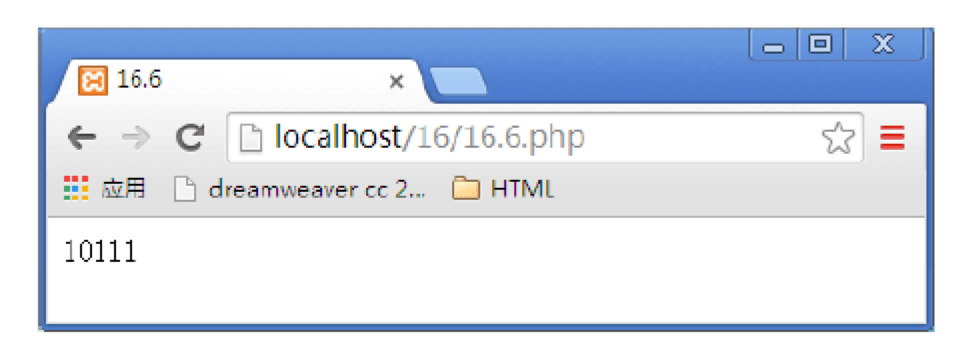


图16.6

16.2.4 PHP变量

变量是计算机编程中一个非常重要的概念。变量可以用来存储数字、文本字符串或数组。简单地说，变量可以表示程序所需的任何信息，而且变量一旦声明，还可以重复使用。在PHP中，变量的声明必须以$符号开始，然后才能声明变量名。

1. 变量命名规范

在PHP中，变量的命名必须要遵循一定的命名规范，具体要求如下所示：

（1）变量名必须以字母或者下划线（_）开头，后面跟上任意字母、数字或者下划线。
（2）变量名不能以数字开头，中间不能有空格及运算符。
（3）变量名严格区分大小写，即$Name与$name 是不同的变量。
（4）为避免命名冲突，不允许使用与 PHP 内置的函数相同的名称。
（5）在为变量命名时，尽量使用有意义的字符串。

2. 变量的赋值

为变量赋值有两种方式：传值赋值和引用赋值。这两种赋值方式在对数据的处理上存在很大差别。

（1）传值赋值：这种赋值方式直接使用一个等号将一个变量或表达式的值赋给另一个变量，这两个变量任何一个值的改变都不会影响到另一个变量的值。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.7</title>
</head>
<body>
<?php
$numA=25;
$numB=$numA;
$numB=33;
echo "变量numA的值为".$numA."<br>";
echo "变量numB的值为".$numB;
?>
</body>
</html>
```

在这段代码中，当执行“$numA=25”时，系统会在内存中为变量numA开辟一个存储空间，并将数值“25”存储在这个空间；当执行“$numB=$numA”时，系统会在内存中为变量numB开辟一个存储空间，并将变量numA所指向的存储空间的内容复制到变量numB所指向的存储空间；当执行“$numB=33”时，系统将变量numB所指向的存储空间保存的值更改为“33”，而变量numA所指向的存储空间保存的值仍然是“25”。运行这段代码后，效果如图16.7所示。

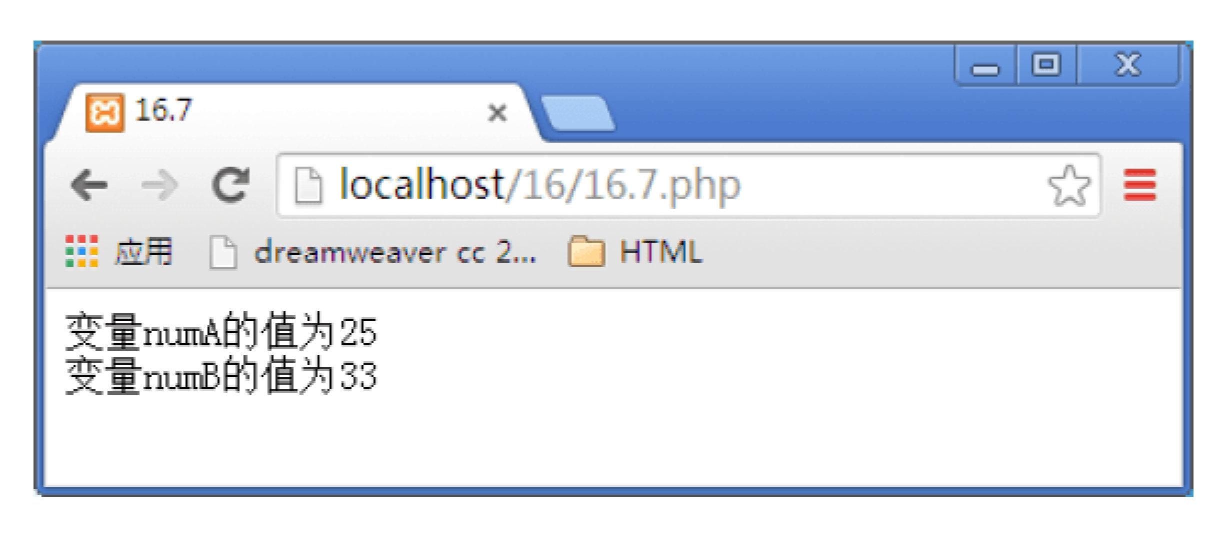


图16.7

（2）引用赋值：引用赋值也是直接使用一个等号将一个变量的值赋给另一个变量，但是等号右边的变量前面需要加上一个“&”符号。使用这种赋值时，并不会为等号左边的变量复制一个新值，而是将等号左边的变量指向等号右边变量指向的内存空间，可以简单的认为这两个变量是同一个存储空间的镜像，任何一个变量值的改变都会影响另一个变量的值。例如下面这段代码：

```
<!doctype html>
```

```
<html>
<head>
<meta charset="utf-8">
<title>16.8</title>
</head>
<body>
<?php
$numA=25;
$numB=&$numA;
$numB=33;
echo "变量numA的值为".$numA."<br>";
echo "变量numB的值为".$numB;
?>
</body>
</html>
```

在这段代码中，因为使用了引用赋值，所以当改变变量numB的值时，变量numA的值也会发生变化。运行这段代码后，效果如图16.8所示。

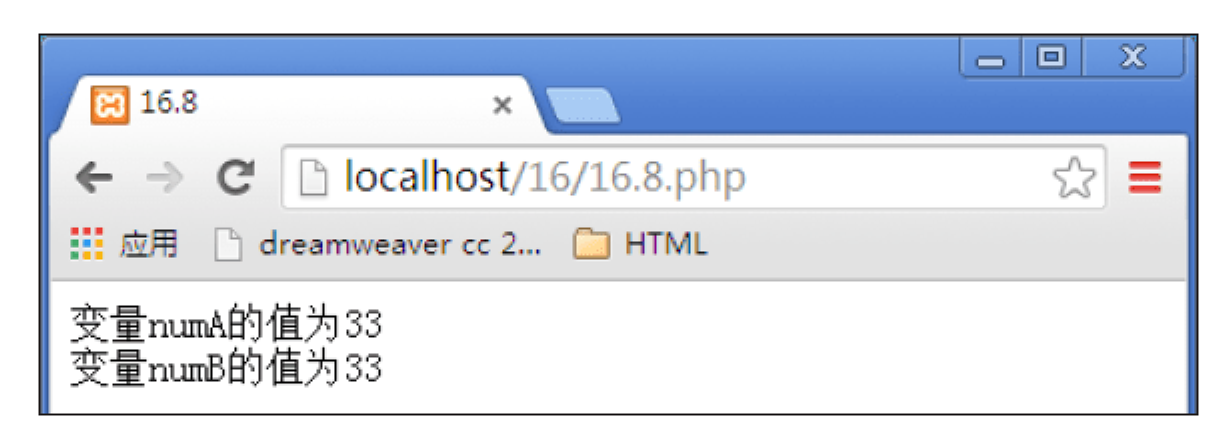


图16.8

3. 变量的作用域

虽然PHP可以在任何位置声明变量，但是变量声明的位置以及方式会决定变量的作用域。所谓变量的作用域，就是指变量在哪些范围内才能被使用。PHP中变量按照作用域的不同可以分为局部变量和全局变量。

（1）局部变量：局部变量是指在函数体内部声明的变量，它的作用域也仅限于函数体内。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.9</title>
</head>
<body>
<?php
function printName(){
$name="张三";
echo "函数内的名字叫".$name."<br>";
}
printName();
$name="李四";
```

```
echo "函数外的名字叫".$name;
?>
</body>
</html>
```

在这段代码中，函数printName内部和外部都声明了一个变量name，但是调用printName函数时输出的是值为张三的内部变量，最后一句输出的是值为李四的变量name，效果如图16.9所示。

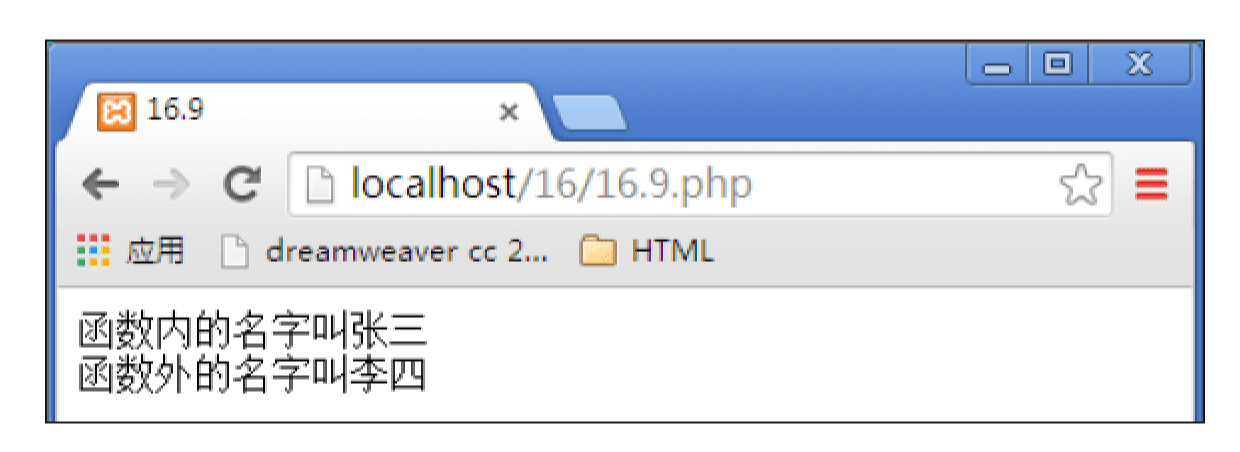


图16.9

（2）全局变量：PHP中声明全局变量只需要在变量前面加上“global”关键字即可，全局变量可以在程序的任何地方访问。如果在函数内部声明全局变量，可以在函数外部引用这个全局变量；如果在函数外部声明全局变量，可以在函数内部引用这个全局变量。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.10</title>
</head>
<body>
<?php
global $name;;
function printName(){
$name="张三";
echo "他的名字叫".$name."。<br>";
}
printName();
function printAge(){
global $age;
$age=25;
}
printAge();
echo "他今年".$age."岁了。";
?>
</body>
</html>
```

在这段代码中，先声明了一个全局变量name，然后在printName函数中为变量name赋值并输出；又在printAge函数中声明了一个全局变量并赋值，然后在函数外面引用了这个变量。运行这段代码后，效果如图16.10所示。

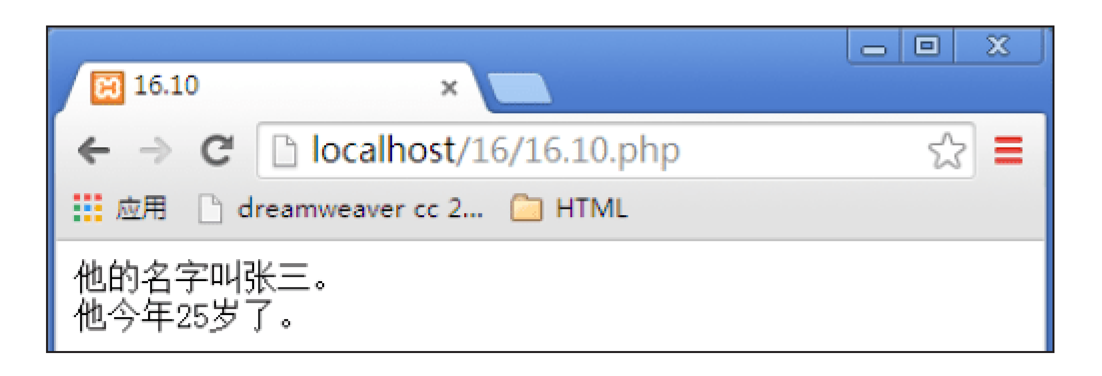


图16.10

（3）静态变量：函数在执行过程中会声明很多变量，在函数执行完成后，这些变量就会消失。如果希望某些函数的值一直保存，即使在下次调用该函数时，仍然保持计算后的变量值，此时就需要用到静态变量。PHP中声明静态变量的方法就是在变量前面添加static关键字。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.11</title>
</head>
<body>
<?php
function add(){
static $num=0;
    $num++;
    echo $num."<br>";
}
add();
add();
add();
?>
</body>
</html>
```

在这段代码中，声明了一个静态变量num，在函数add中变量num会自动加1，然后输出。将这个函数调用3次，因为使用了静态变量，所以每次调用add函数时，变量num的值总保持最后一次计算的结果。运行这段代码后，效果如图16.11所示。

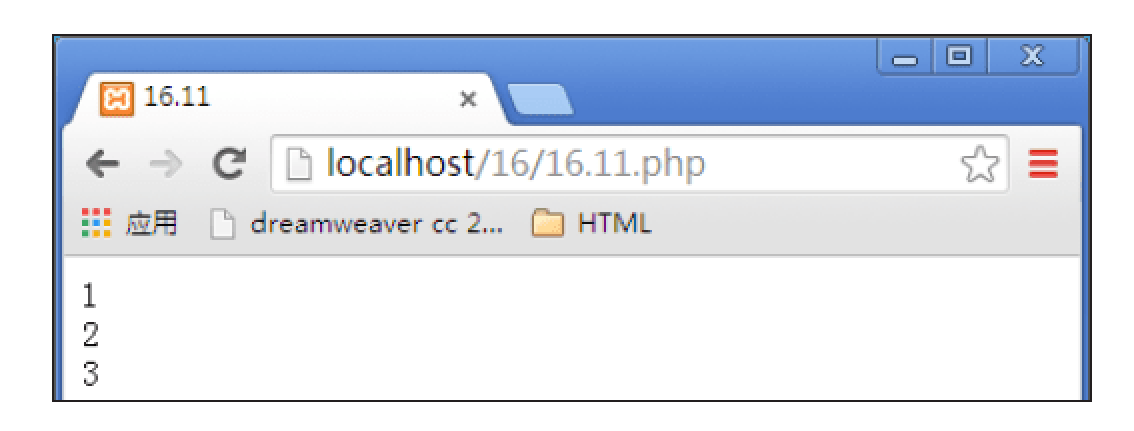


图16.11

（4）可变变量：可变变量是一种独特的变量，它可以动态地改变一个变量的名称，方法就是在该变量的前面加一个变量符号“$”。例如下面这段代码：

```
<!doctype html>
<html>
```

```
<head>
<meta charset="utf-8">
<title>16.12</title>
</head>
<body>
<?php
 $str= "hello";
 $$str = "world";
 echo $str."<br>";
 echo $$str."<br>";
 echo $hello."<br>";
 echo "$str {$$str}"."<br>";
 echo "$str $hello";
?>
</body>
</html>
```

在这段代码中，第1行是一个普通变量，第2行是一个可变变量，相当于声明了一个名为hello的变量，即$hello=“world”。下面5个输出，第1个是正常输出变量，第2个是正常输出可变变量，第3个是输出与可变变量等价的变量，第4个是输出普通变量与可变变量，最后一个是输出普通变量和等价于可变变量的变量。运行这段代码后，效果如图16.12所示。

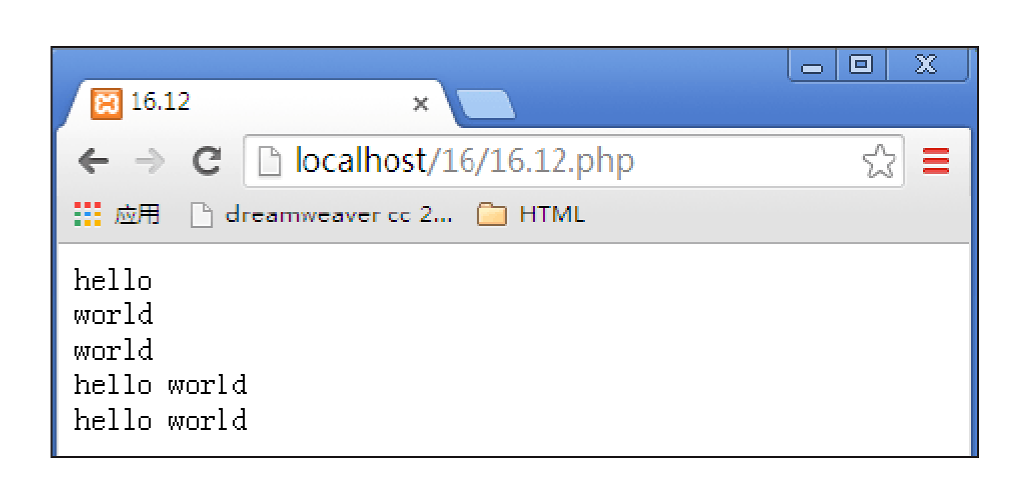


图16.12

（5）预定义变量：预定义变量又称为超级全局变量数组，是PHP系统中自带的变量，不需要开发者重新定义，它可以让你的程序设计更加方便快捷。在PHP脚本运行时，PHP会自动将一些数据放在超级全局变量数组中。PHP中预定义变量如表16.2所示。

表16.2

变量	作用
$GLOBALS[]	存储当前脚本中的所有全局变量，其KEY为变量名，VALUE为变量值
$_SERVER[]	当前WEB服务器变量数组
$_GET[]	存储以GET方法提交表单中的数据
$_POST[]	存储以POST方法提交表单中的数据
$_COOKIE[]	取得或设置用户浏览器Cookies中存储的变量数组
$_FILES[]	存储上传文件提交到当前脚本的数据
$_ENV[]	存储当前WEB环境变量
$_REQUEST[]	存储提交表单中所有请求数据，其中包括$_GET、$_POST、$_COOKIE和$_SESSION中的所有内容
$_SESSION[]	存储当前脚本的会话变量数组

4. 变量的数据类型

数据类型是具有相同特性的一组数据的统称。PHP早就提供了丰富的数据类型，PHP 5 中又有更多补充。数据类型可以分为 3类：标量数据类型、复合数据类型和特殊数据类型。 标量类型包括整型（int和integer）、浮点型（float、double和real）、布尔型（bool、boolean）和字符串（string），复合类型包括数组（array）和对象（object），特殊类型包括资源（resource）和空值（NULL）。

5. 变量类型的转换

PHP中的类型转换包括两种方式，即自动类型转换和强制类型转换。

（1）自动类型转换：自动类型转换是指在定义变量时不需要指定变量的数据类型，PHP会根据引用变量的具体应用环境将变量转换为合适的数据类型。如果所有运算数都是数字，则将选取占用字节最长的一种运算数的数据类型作为基准数据类型；如果运算数为字符串，则将该字符串转换为数字然后再进行求值运算。字符串转换为数字的规定为如果字符串以数字开头，则只取数字部分而去除数字后面部分，根据数字部分构成决定转换为整型数据还是浮点型数据；如果字符串以字母开头，则直接将字符串转换为 0。例如下面这段代码，运行后的效果如图16.13所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.13</title>
</head>
<body>
<?php
$numA=1+1.25;
$numB=2+"2.25";
$numC=3+"abc";
echo $numA."<br>";
echo $numB."<br>";
echo $numC."<br>";
?>
</body>
</html>
```

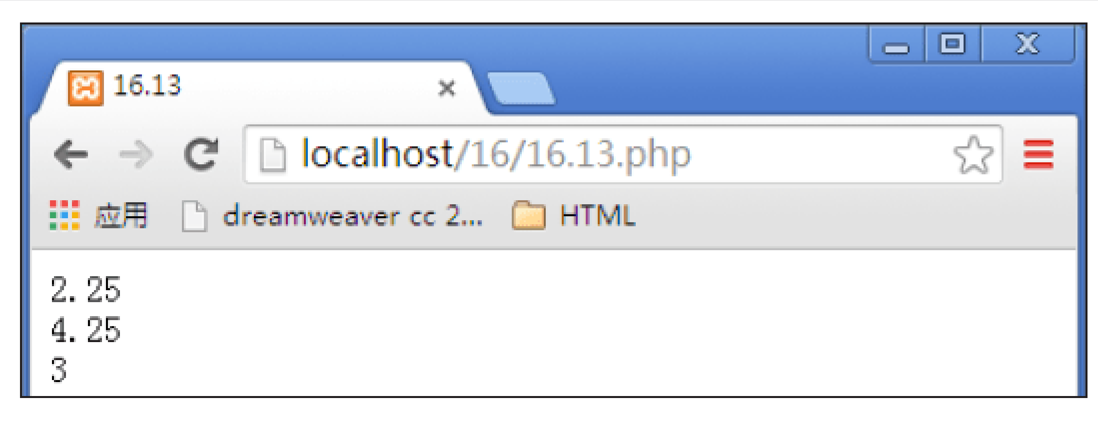


图16.13

（2）强制类型转换：强制类型转换允许我们手动将变量的数据类型转换为指定的数据类型。PHP强制类型转换与 C 语言或者 Java 语言中的类型转换相似，都是通过在变量前面加上一个小括号，并把目标数据类型填写在括号中来实现的。例如下面的代码，运行后的效果如图16.14所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.14</title>
</head>
<body>
<?php
$numA="5";
$numB=(int)$numA;
$numC=(bool)$numA;
$numD=(float)$numA;
$numE=(double)$numA;
$numF=(real)$numA;
$numG=(string)$numA;
$numH=(array)$numA;
echo "原始数据：".$numA."<br>";
echo "转换为整型：".$numB."<br>";
echo "转换为布尔型：".$numC."<br>";
echo "转换为浮点型：".$numD."<br>";
echo "转换为双精度：".$numE."<br>";
echo "转换为real型：".$numF."<br>";
echo "转换为字符串：".$numG."<br>";
echo "转换为数组：".$numH."<br>";
?>
</body>
</html>
```

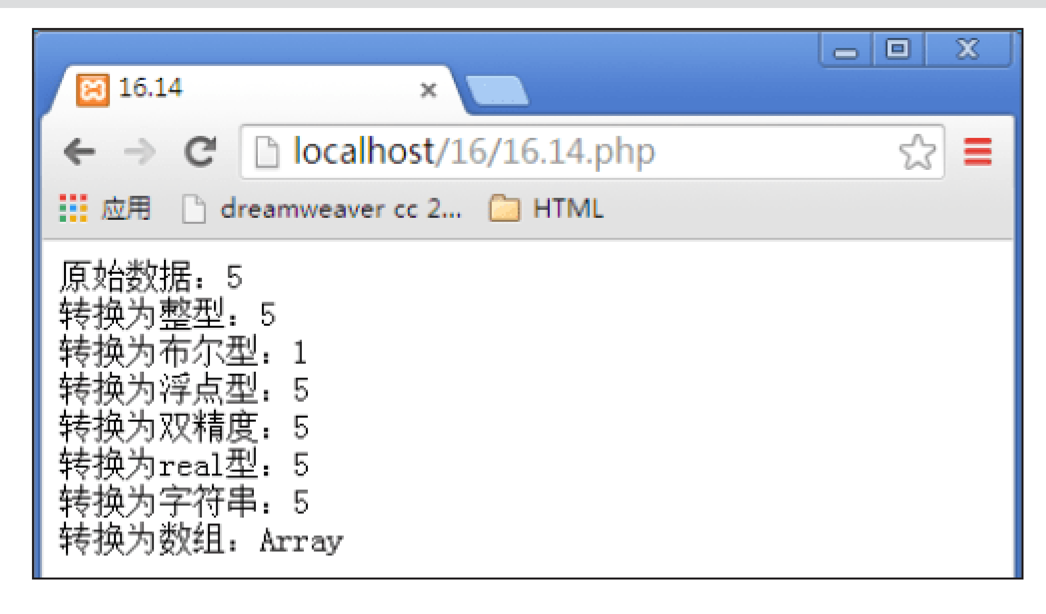


图16.14

6. 变量的常用函数

PHP中含有很多系统定义的变量操作常用函数，这些函数包括以下几种。

（1）变量转换函数：除了强制类型转换以外，还可以通过变量转换函数对变量进行转换。例如，使用settype函数将变量转换为指定的类型，示例代码如下所示：

```
<?php
 $a = "1.75m";
 $b = true;
 settype($a,"float")."<br>";
 settype($b,"string")."<br>";
 echo $a."<br>";
 echo $b;
```

```
?>
```

变量a输出结果为1.75，变量b输出结果为1。另外，intval函数用于将变量转换为整数，floatval函数用于将变量转换为浮点数，strval函数用于将变量转换为字符串。这3个函数的用法如下所示：

```
<?php
$a = "1.75m";
$b = 2015;
$c = intval($a);
$d = floatva($a);
$e = strval($a);
echo $a."<br>";
echo $b."<br>";
echo $c."<br>";
echo $d."<br>";
echo $e."<br>";
?>
```

（2）变量检查函数：这类函数用于对变量进行检查。例如，使用isset函数检查某个变量是否存在，如果存在则返回true，否则返回false；使用empty函数检查某个变量的值是否为空，如果为空则返回true，否则返回false。相关示例代码如下所示：

```
<?php
$a = "1.75m";
$b = "";
echo isset($a)."<br>";
echo isset($b)."<br>";
echo isset($c)."<br>";
echo empty($a)."<br>";
echo empty($b)."<br>";
?>
```

（3）变量判断函数：PHP 中有一些函数可以判断变量的类型，这些函数如表16.3所示。

表16.3

函数名	作用
is_int()、is_integer()	检测变量是否为整数
is_float()、is_double()	检测变量是否为浮点数
is_bool()	检测变量是否为布尔
is_string()	检测变量是否为字符串
is_array()	检测变量是否为数组
is_object()	检测变量是否为一个对象
is_resource()	检测变量是否为资源类型
is_null()	检测变量是否为NULL

16.2.5 PHP常量

常量是一种特殊的变量，它的变量值声明之后将不会改变。PHP中有两个常量，分别为自定义常量和预定义常量。

1. 自定义常量

PHP中使用define函数定义常量。常量命名方法与变量命名相同，以字母或下划线开头。常量名称区分大小写，但按照惯例常量名称全部大写。如果设置为 true则不区分大小写，如果设置为false则区分大小写，如果没有设置该参数，则取默认值false。一个常量一旦被定义，就不能再改变或者取消定义。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.15</title>
</head>
<body>
<?php
define("SYSTEM_NAME","哈哈");
echo SYSTEM_NAME."<br>";
define("SYSTEM_NAME","啦啦");
echo SYSTEM_NAME."<br>";
echo System_Name;
?>
</body>
</html>
```

在这段代码中，第1次定义的常量正确输出；第2次定义的常量应与第一次定义的常量同名，因此并不更改它的值，依然输出第一次赋的值；最后一个输出的常量名没有区分大小写，被认为是一个没有定义的常量。运行这段代码后，效果如图16.15所示。

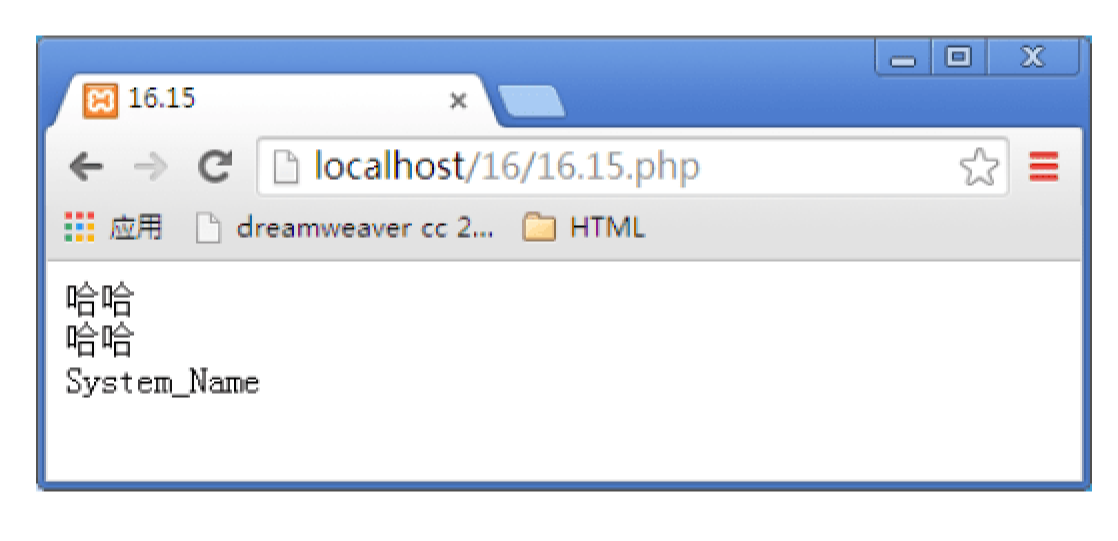


图16.15

2. 预定义常量

PHP也提供了一些默认的预定义常量供使用。在程序中可以随时应用这些预定义常量，但是我们不能任意更改这些常量的值。PHP中常用的一些默认预定义常量及其作用如表16.4所示。

表16.4

函数名	作用
FILE	存储当前脚本的绝对路径及文件名称
LINE	存储该常量所在的行号
FUNCTION	存储该常量所在的函数名称
CLASS	存储该常量所在的类的名称
METHOD	存储该常量所在的类的方法名称
PHP_VERSION	存储当前PHP的版本号
PHP_OS	存储当前服务器的操作系统

16.2.6 运算符

一个复杂的PHP程序往往是由大量的表达式所构成的，而运算符则是表达式的核心，也称作操作符。只有掌握了PHP表达式和运算符的用法，才能够更好地使用PHP语言进行开发设计。PHP中常用的运算符包括算术运算符、赋值运算符、比较运算符、逻辑运算符、位运算符、字符串运算符和数组运算符，下面将分别介绍。

1. 算术运算符

算术运算符主要用于处理加减乘除和取模运算。相关代码如下所示，运行结果如图16.16所示。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.16</title>
</head>
<body>
<?php
 $a = 22;
 $b = 6;
 echo "加运算结果是".($a+$b)."<br>";
 echo "减运算结果是".($a-$b)."<br>";
 echo "乘运算结果是".$a*$b."<br>";
 echo "除运算结果是".$a/$b."<br>";
 echo "取模运算结果是".$a%$b;
?>
</body>
</html>
```

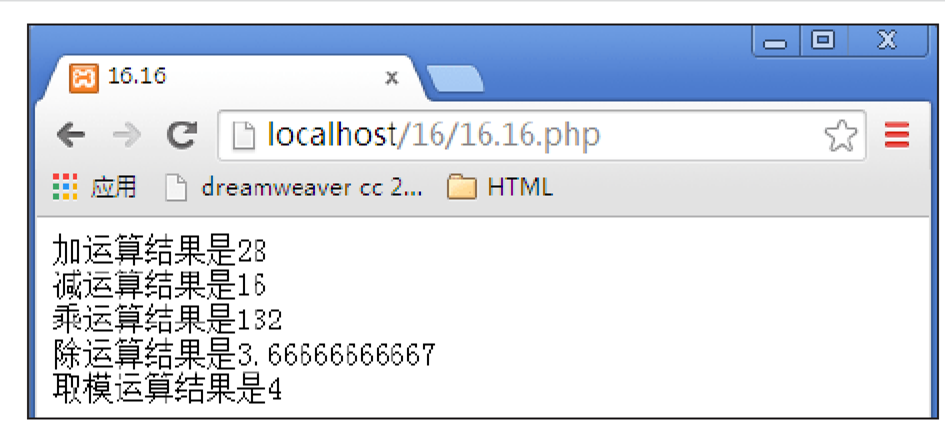


图16.16

2. 递增/递减运算符

递增/递减运算符是可以对操作系统（可以是数字或字符）进行递增、递减操作的一种运算符。PHP中有4种这样的操作符，第1种如$i++，表示返回$i，然后将$i的值加1；第2种如++$i，表示$i的值加1，然后返回$i；第3种如$i--，表示返回$i，然后将$i的值减1；第4种如--$i，表示$i的值减1，然后返回$i。

3. 赋值运算符

最常用的赋值运算符就是等号，它的实际意义是将等号右边的值赋给等号左边的变量。PHP中还提供了更多的赋值运算符，例如表16.5所示。

表16.5

赋值运算符	用法	等价格式
=	$i=5	$i=5
+=	$i+=5	$i=$i+5
-=	$i-=5	$i=$i-5
*=	$i*=5	$i=$i*5
/=	$i/=5	$i=$i/5
%=	$i%=5	$i=$i%5
.=	$i.= "abc"	$i=$i. "abc"

4. 比较运算符

比较运算符也称条件运算符或关系运算符，用于比较两个数据的值并返回一个布尔类型的结果。PHP 提供的比较运算符及其用法如表16.6所示。

表16.6

比较运算符	名称	用法	说明
==	等于	$a==$b	如果变量a等于变量b，则返回true
===	全等	$a===$b	如果变量a等于变量b，并且他们的数据类型也相同，则返回true
!=或<>	不等	$a!=$b或$a<>$b	如果变量a不等于变量b，则返回true
!==	非全等	$a!==$b	如果变量a不等于变量b，则返回true
<	小于	$a<$b	如果变量a小于变量b，则返回true
>	大于	$a>$b	如果变量a大于变量b，则返回true
<=	小于等于	$a<=$b	如果变量a小于或等于变量b，则返回true
>=	大于等于	$a>=$b	如果变量a大于或等于变量b，则返回true

5. 逻辑运算符

逻辑运算符用于处理逻辑运算操作，只能操作布尔型值。PHP提供的逻辑运算符及其用法如表16.7所示。

表16.7

逻辑运算符	名称	用法	说明
and或&&	逻辑与	$a and $b或$a && $b	如果$a和$b两个都为true时返回true
or或‖	逻辑或	$a or $b或$a ‖ $b	如果$a和$b任何一个为true时返回true
xor	逻辑异或	$a xor $b	如果$a和$b只有一个为true时返回true
!	逻辑非	!$a	如果$a为false时返回true

6. 位运算符

位运算符主要应用于整型数据的运算过程。当表达式包含位运算符时，运算时会先将各个整型运算数转换为相应的二进制格式，然后再进行位运算。PHP提供的位运算符及其用法如表16.8所示。

表16.8

逻辑运算符	名称	用法	说明
&	与操作符	$a & $b	将在$a和$b中都为1的位设为1
|	或操作符	$a | $b	将在$a或者$b中为1的位设为1
^	异或操作符	$a ^ $b	将在$a和$b中不同的位设为1
~	非操作符	~$a	将$a中为0的位设为1,为1的位设为0
<<	左移操作符	$a<<2	将$a中的位向左移动2次，空出的位置置为0，每一次移动都表示乘以2
>>	右移操作符	$a>>$2	将$a中的位向右移动2次，多出的位置截掉，每一次移动都表示乘以2

7. 字符串运算符

字符串运算符也称连接运算符，用于处理字符串的相关操作。在PHP中提供了两个字符串运算符。第1个是连接运算符（.），它返回左右参数连接后的字符串。第2个是连接赋值运算符（.=），它将右边的参数附加到左边的参数后。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.17</title>
</head>
<body>
<?php
$a="我叫";
$b=$a."张三";
echo $b."<br>";
$b.="，不叫李四。";
echo $b;
?>
</body>
</html>
```

在这段代码中，为变量a赋值一个字符串，再将变量a和另一个字符串连接后赋值给变量b，然后输出字符串b，再通过.=运算符将变量b与第3个字符串连接，最后输出的效果如图16.17所示。

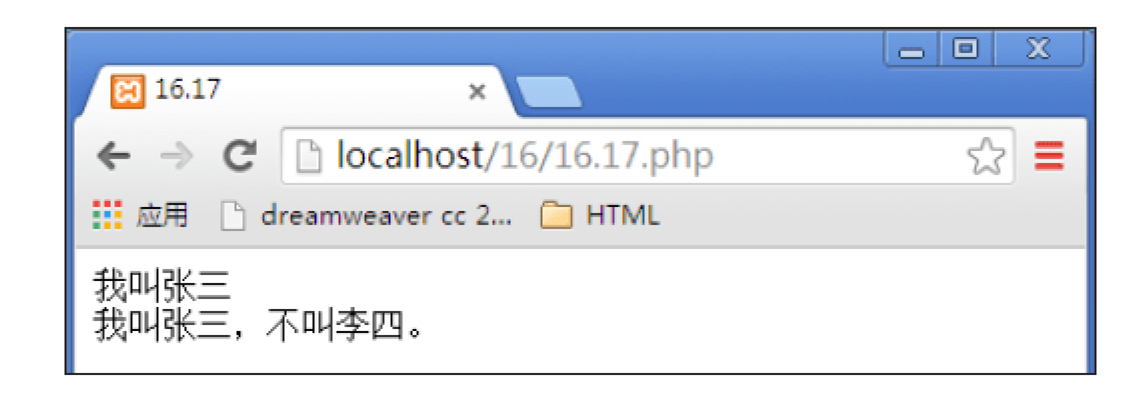


图16.17

8. 数组运算符

数组运算符应用于数组的一些相关操作。PHP 提供的数组运算符及其用法如表16.9所示。

表16.9

逻辑运算符	名称	用法	说明
+	联合	$a+$b	$a与$b保存的数组联合
==	相等	$a==$b	如果$a与$b保存的数组具有相同键值，则返回true
===	全等	$a===$b	如果$a与$b保存的数组具有相同的键值，且顺序和数据类型一致则返回true
!=或<>	不等	$a!=$b或$a<>$b	如果$a与$b保存的数据不具有相同的键值，则返回true
!==	不全等	$a!==$b	如果$a与$b保存的数组不具有相同键值，且顺序和数据类型也不一致，则返回true

9. 错误抑制运算符

当PHP表达式产生错误而我们又不想将错误信息显示在页面上时，可使用错误抑制运算符；当表达式的前面被加上“@”这个运算符以后，该表达式可能产生的任何错误信息都会被忽略。例如运行下面这段代码将会得到如图16.18所示的错误。

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.18</title>
</head>
<body>
<?php
$numA=15;
$numB=0;
echo $numA/$numB;
?>
</body>
</html>
```

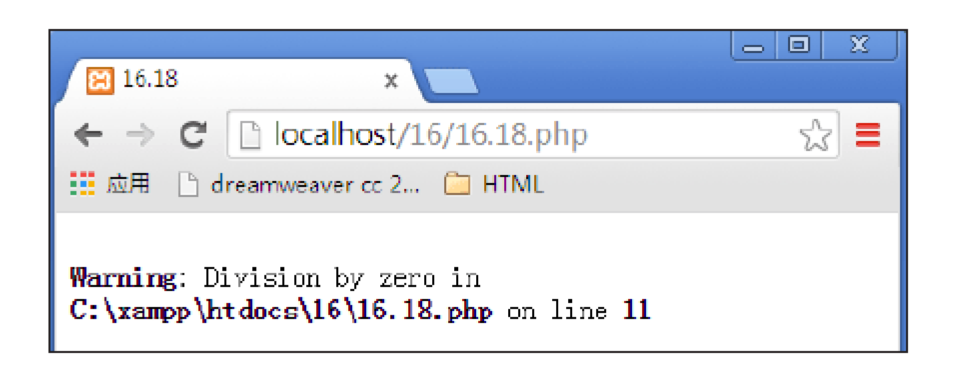


图16.18

如果在运算符前面添加@，例如下面这段代码，刷新页面后就不会出现任何错误信息。这种处理错误的方法在发布程序的时候会用到，用以避免页面上出现意想不到的错误信息。

```
echo @($numA/$numB);
```

10. 执行运算符

执行运算符使用“`”（键盘数字1左边的按键）符号。使用了这个运算符以后，该运算符内的字符串将会被当做 DOS命令行来处理。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.19</title>
</head>
<body>
<?php
$path="dir C:\\xampp\\htdocs";
echo $path;
?>
</body>
</html>
```

运行这段代码后，效果如图16.19所示。

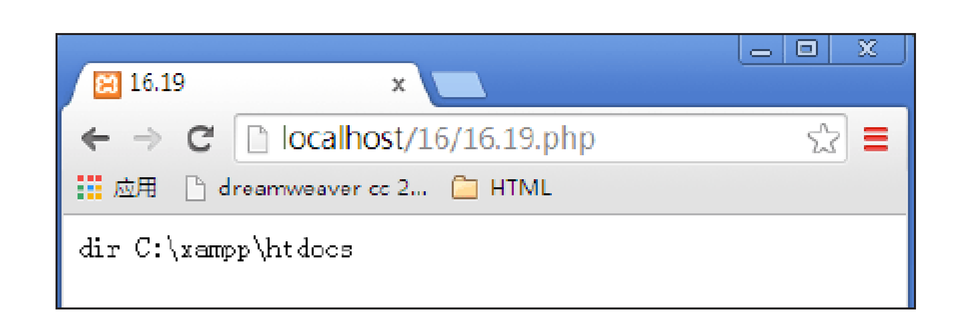


图16.19

11. 三元运算符

三元运算符的功能与if....else流程语句相同，它在一行中书写，代码精练、执行效率高。在PHP程序中恰当地使用三元运算符能够让脚本更为简洁、高效。三元运算符的语法如下所示：

```
表达式1?表达式2:表达式3
```

这句话的意思是，当表达式1的值为true时，运算结果为表达式2的值，否则运算结果是表达式3的值。例如下面这段代码：

```
<!doctype html>
<html>
```

```
<head>
<meta charset="utf-8">
<title>16.20</title>
</head>
<body>
<?php
$fristName="Sean";
$lastName=$fristName=="Sean"?"Zhao":"Li";
echo $fristName." ".$lastName;
?>
</body>
</html>
```

在这段代码中，首先给变量fristName赋值为“Sean”，然后判断变量fristName的值是否等于“Sean”，返回结果为true，所以给变量lastName赋值“Zhao”，最后输出两个变量，运行效果如图16.20所示。

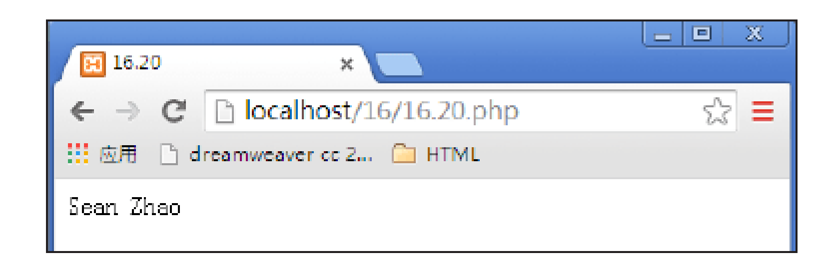


图16.20

12. 运算符的优先级

一个复杂的表达式往往包含了多种运算符，各个运算符优先级的不同决定了其被执行的顺序也不一样。高优先级的运算符所在的子表达式会先被执行，而低优先级的运算符所在的子表达式会后被执行。下面表格从高到低列出了PHP运算符的优先级。同一行中的运算符具有相同的优先级，它们结合的方向决定求值顺序。左联表示表达式从左向右求值，右联相反。

表16.10

结合方向	运算符	附加信息
无方向性	new	new
左	[	arrar()
无方向性	++-	递增/递减运算符
无方向性	!~-(int)(float)(string)(array)(object)@	类型
左	*/%	算术运算符
左	+-	算术运算符和字符串运算符
左	<<>>	位运算符
无方向性	< <= > >=	比较运算符
无方向性	== != === !==	比较运算符
左	&	位运算符和引用
左	^	位运算符
左	|	位运算符
左	&&	逻辑运算符
左	||	逻辑运算符
左	?:	三元运算符

结合方向	运算符	附加信息
右	= += -= *= /= .= %= &= |= ^= <<= >>=	赋值运算符
左	and	逻辑运算符
左	xor	逻辑运算符
左	or	逻辑运算符

在创建复杂的表达式时，还可以使用圆括号来限制运算符的优先级，让优先级低的表达式先进行运算，再进行其他处理。



16.3 流程控制语句

任何一种编程语言，要实现复杂的功能，都必须使用流程控制语句将多条语句组成一个程序，PHP语言也不例外。下面我们就介绍一下PHP语言中的分支语句和循环语句，以及特殊的流程控制语句。

16.3.1 分支语句

分支语句是一种非常常见的流程控制语句，它表示非此即彼的流程选择，这类流程控制语句可以分为以下几种。

1. if语句

if语句是一种单条件的判断语句，它的语法规则如下：

```
if (expr) statement
```

它表示如果expr表达式的条件成立，那么执行statement这条语句。如果有多个statement语句，可以用大括号括起来，例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.21</title>
</head>
<body>
<?php
$a=5;
$b=10;
if($b>$a){
echo "a=".$a."<br>";
echo "b=".$b."<br>";
```

```
echo "b大于a。";
}
?>
</body>
</html>
```

运行这段代码后，效果如图16.21所示。

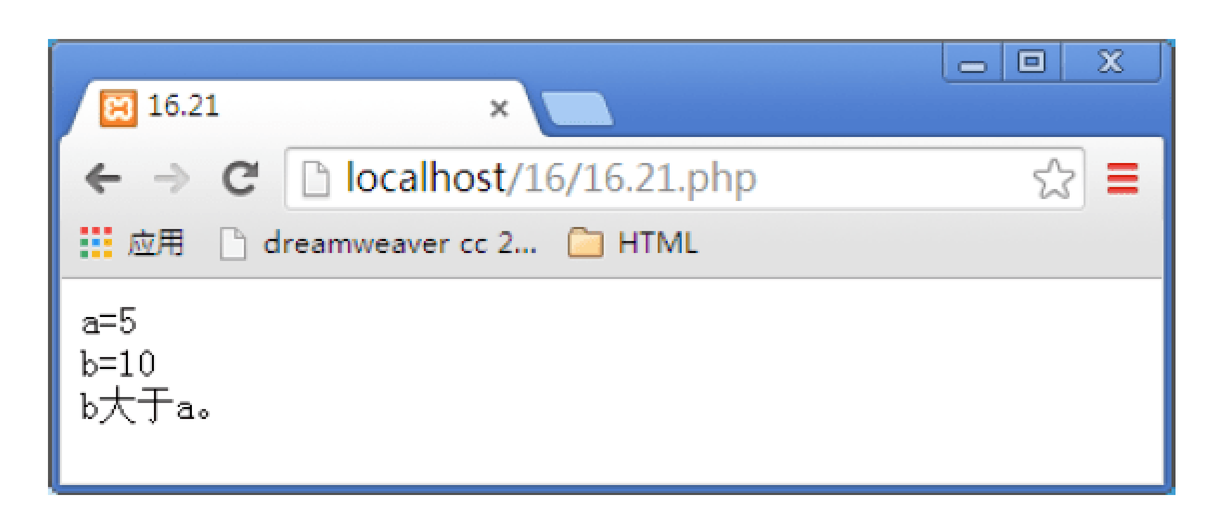


图16.21

另外，statement语句中还可以再嵌套if语句。

2. else语句

else语句必须与if语句配合使用，表示如果expr表达式返回false时，应该执行的语句。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.22</title>
</head>
<body>
<?php
$a=5;
$b=10;
if($b>$a){
echo "b大于a。";
}else{
echo "a大于b。";
}
?>
</body>
</html>
```

运行这段代码后，页面上将输出“b大于a。”。

3. else if语句

else if语句表示当if语句返回false时，通过新的条件来控制流程。它与else不同，else仅当if返回为false时才可以执行，而else if将根据新的条件来控制后面的流程。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.23</title>
</head>
<body>
<?php
$num=23;
if($num%5==1){
echo "余数是1<br>";
}else if($num%5==2){
echo "余数是2<br>";
}else if($num%5==3){
echo "余数是3<br>";
}else{
echo "不会算。<br>";
}
?>
</body>
</html>
```

在这段代码中，第一个if语句中的表达式不成立，继续执行下一个 else if语句，这个语句的表达式仍然不成立，继续执行下一个else if语句，这个语句的表达式成立，所以输出信息为“余数是3”。在最后一个else if语句的后面，一般会再跟一个else语句，控制前面所有条件都不成立时的流程。

4. switch语句

switch语句可以看做是将多个if语句组合成一个新的控制语句，根据表达式的值选择不同的执行语句。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.24</title>
</head>
<body>
<?php
$num=2;
switch($num){
case 0:
case 1:
print "变量的值等于0或者1。";
break;
case 2:
print "变量的值等于2。";
break;
default:
print "还不能确定变量的值。";
```

```
break;
}
?>
</body>
</html>
```

在这段代码中，switch语句根据变量num的值选择要执行的语句。如果变量num的值是0或者1，将执行输出“变量的值等于0或者1。”的语句；如果变量num的值是2，将执行输出“变量的值等于2。”的语句；否则将执行输出“还不能确定变量的值。”的语句。

对于switch语句而言，如果满足多个条件时，可执行相同的语句，那么这些条件可以写在一起，如本例中的0和1。每个case语句的后面都需要一个break语句，这是switch语句的规则。switch的最后一种情况，需要使用一个default分支，用于处理以上表达式都不成立的情况。

16.3.2 循环语句

循环语句是重复执行一个或多个语句。PHP语言中循环语句主要有以下3种。

1. while语句

while语句表示先判断表达式的条件，然后再执行循环体的语句。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.25</title>
</head>
<body>
<?php
$num=0;
$sum=0;
while($num<=10){
$sum+=$num;
$num++;
}
print "结果为".$sum;
?>
</body>
</html>
```

在这段代码中，先声明了变量num和sum，在while表达式中判断num是否小于等于10，如果表达式成立，那么将变量sum和num相加的结果赋值给变量sum，然后将num的值加1，继续执行下一次循环，这样就能计算出从0递加到10的总和。

2. do…while语句

do…while语句与while语句相同，但是while语句先判断表达式，然后再执行语句，而do…while语句则是先执行循环的语句，再判断表达式，如果表达式成立，那么继续执行下一

次循环，否则终止循环。例如下面的代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.26</title>
</head>
<body>
<?php
$num=10;
do{
print $num."<br>";
$num--;
}while($num>5)
?>
</body>
</html>
```

在这段代码中，先声明一个变量num，然后赋值为10，执行循环语句，输出变量，并将变量递减，最后判断变量的值是否大于5，如果大于5则继续下一次循环，否则终止循环。运行这段代码后，页面效果如图16.22所示。

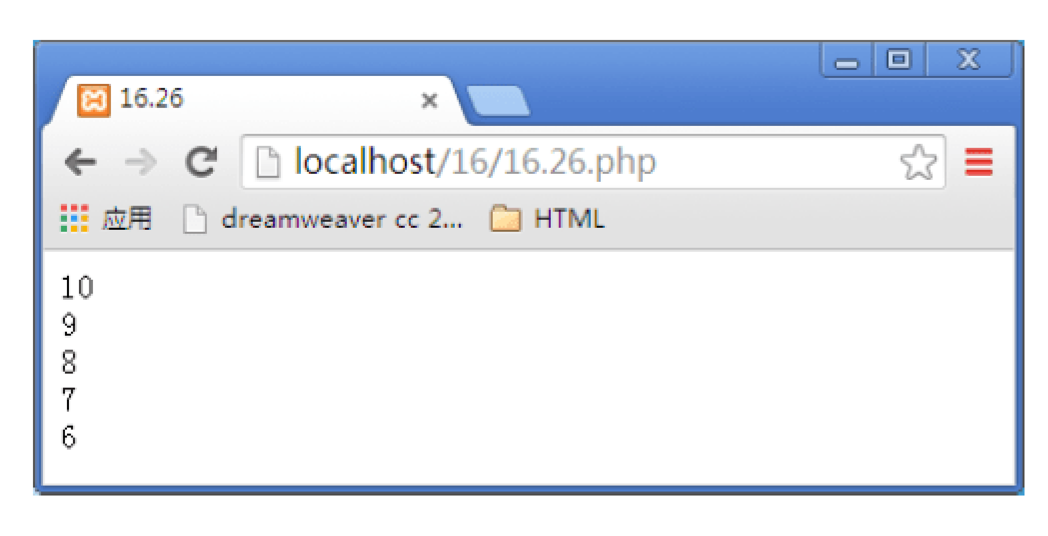


图16.22

3. for语句

以上两种循环都是在循环次数不确定的情况下使用的循环，如果已经确定需要循环多少次，可以直接使用for循环语句。for循环语句的语法如下所示：

```
for(expr1; expr2; expr 3) statement
```

第一个表达式expr1始终都会无条件的在循环开始的时候执行；每一次循环的时候，表达式expr2都会被执行，如果结果为true，则继续执行循环，否则结束循环；每次循环结束后，都会计算表达式expr3。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.27</title>
</head>
<body>
```

```
<?php
for($i=0;$i<10;$i++){
print $i."-";
}
?>
</body>
</html>
```

运行这段代码后，效果如图16.23所示。

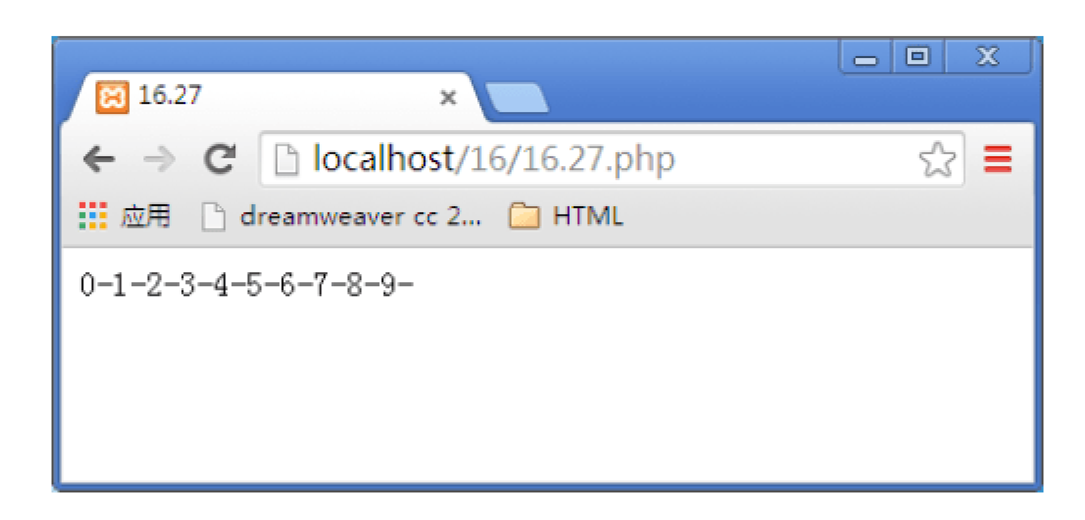


图16.23

在for循环中，3个表达式都可以为空，但是不建议将第2个表达式省略，因为那样的话将无法确定循环的次数，从而进入无限循环中。

16.3.3 特殊流程控制

为了更好的控制循环语句，在某些特殊情况下，需要使用特殊的流程控制语句，以满足各种需求。

1. break

用于中断当前执行，可以在任意的循环语句中使用。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.28</title>
</head>
<body>
<?php
for($i=0;$i<10;$i++){
if($i>5){
break;
}
print $i."-";
}
?>
</body>
</html>
```

在这段代码中，for循环用于输出0到9，而if语句控制当变量i大于5时，执行break语句，

中断所有操作。运行这段代码后，效果如图16.24所示。

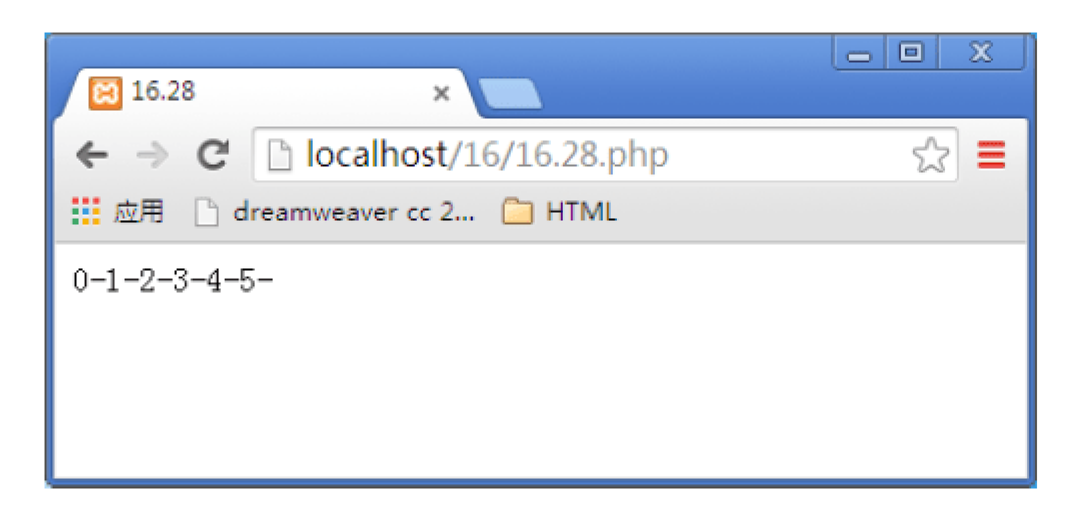


图16.24

2. continue

continue语句只能在循环语句的内部使用，用于跳过这次循环，继续执行下一次循环。例如下面这段代码：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>16.29</title>
</head>
<body>
<?php
for($i=0;$i<10;$i++){
if($i%2==0){
continue;
}
print $i."-";
}
?>
</body>
</html>
```

在这个循环语句中，如果变量对2求余等于0，也就是说，如果变量是偶数，将不执行输出语句，所以输出的都是奇数，效果如图16.25所示。

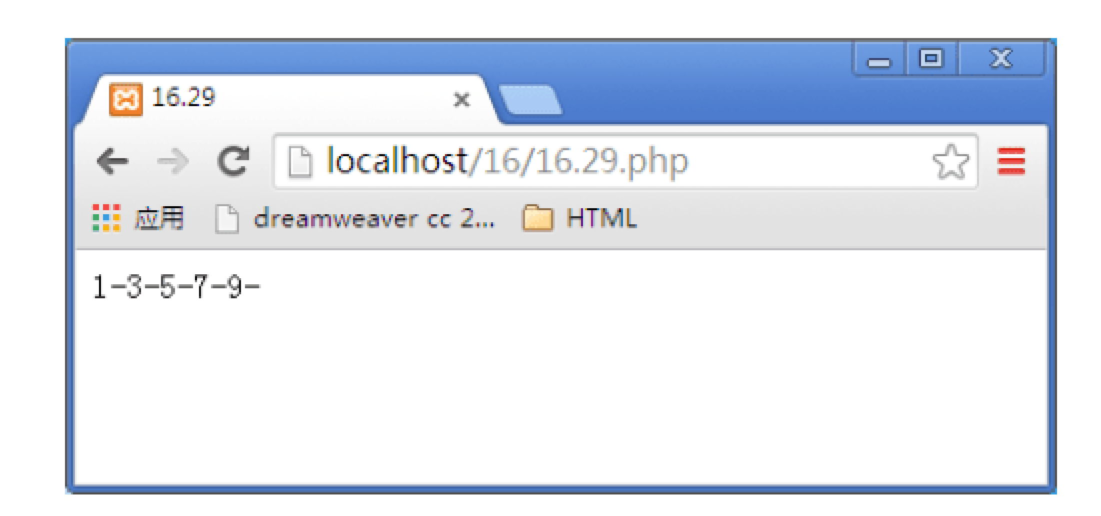


图16.25

第17章 使用MySQL数据库

数据库是存放数据的仓库，数据库中的数据具有一定的关联关系，这些数据可以是文本、图像、音频或视频。MySQL数据库是一种关系型数据库，这种数据库是由一个或多个表格组成的，每个表格中存放的就是数据。



17.1 Windows下安装和配置MySQL数据库

MySQL数据库是一个免费的数据库，它可以在多个平台上安装和运行，下面我们以Windows环境为例，介绍MySQL数据库的安装和配置方法。

17.1.1 下载与配置免安装版本

用户可以到MySQL数据库官网上下载安装或免安装的版本，这里以免安装的mysql-noinstall-5.1.73-winx64.zip为例进行讲解，用户可以到以下网址下载。

```
http://dev.mysql.com/downloads/mysql/5.1.html#downloads
```

下载成功后，可以按照以下操作步骤进行配置：

01 将mysql-noinstall-5.1.73-winx64.zip压缩包解压到C:\Program Files文件夹下。

02 打开解压后的文件夹，找到名为“my-small.ini”的配置文件，将其重命名为“my.ini”，然后用记事本打开该文件，在[client]和[mysqld]下均添加一行：default-character-set=gbk。

03 打开Windows环境变量设置，新建变量名MYSQL_HOME，变量值为MySQL的安装目录，即C:\Program Files\ mysql-noinstall-5.1.73-winx64。

04 在环境变量的Path变量中添加“;%MYSQL_HOME%\bin;”，这里注意不要遗忘了前后的分号。

05 安装MySQL服务，打开Windows命令提示符，执行以下命令：

```
mysqld -install MySql -defaults-file="my.ini"
```

如果提示“Service successfully installed.”则表示安装成功。然后打开Windows命令提示符，如果要启动MySQL，可以输入net start MySQL；如果要停止MySQL，可以输入net stop

MySQL；如果要卸载MySQL，可以输入sc delete MySQL。

原生态的MySQL数据库操作需要使用Windows命令提示符，操作非常不方便，为了提高工作效率，可以使用其他可视化的操作界面来维护MySQL数据库。

17.1.2 通过安装XAMPP安装MySQL数据库

XAMPP是一个把Apache网页服务器与PHP、Perl以及MySQL集合在一起的一个免费的安装包，它允许用户在自己的电脑上轻易的建立网页服务器。下面我们介绍如何安装和配置MySQL数据库。

用户可以到官网（http://www.xampps.com/）下载XAMPP的安装包，直接点击安装，安装过程会提示用户选择安装相应的服务，此时可以勾选Apache服务和MySQL服务。安装完成后，点击桌面的快捷方式启动XAMPP，启动后看到如图17.1所示图形表示安装成功。如果Apache服务不能启动，请检查您是否安装过IIS，因为Apache的默认端口是80，与IIS端口冲突，如果冲突，请打开xampp/apache/conf/httpd.conf文件，把listen 80修改为其他的端口。

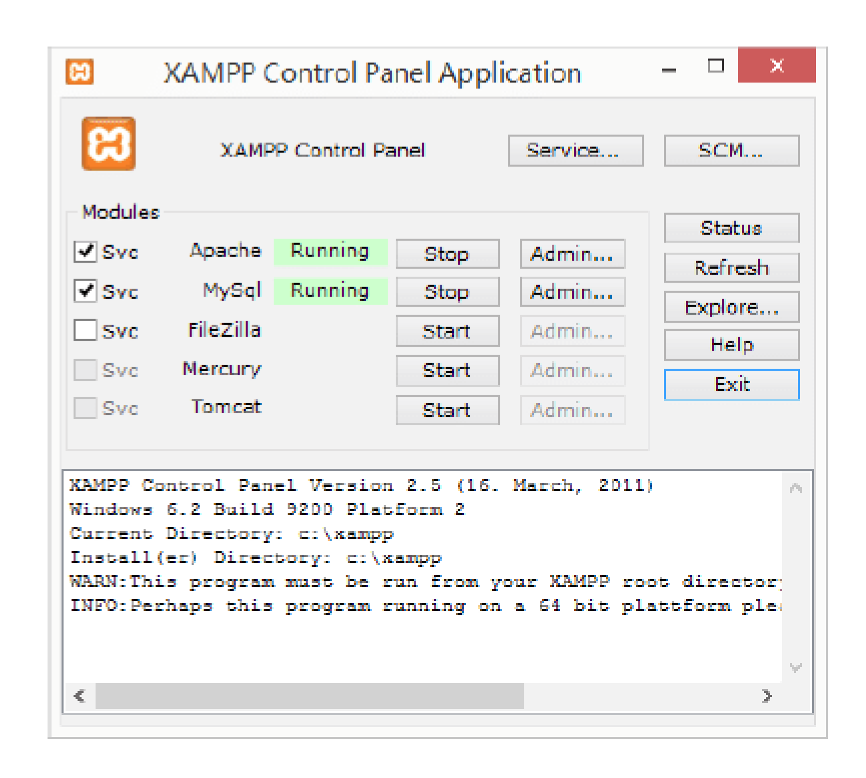


图17.1

在这个界面中，单击MySQL后面的admin按钮，可以打开phpMyAdmin操作界面，在这个操作界面中可以对MySQL数据库进行各种可视化操作，如图17.2所示。

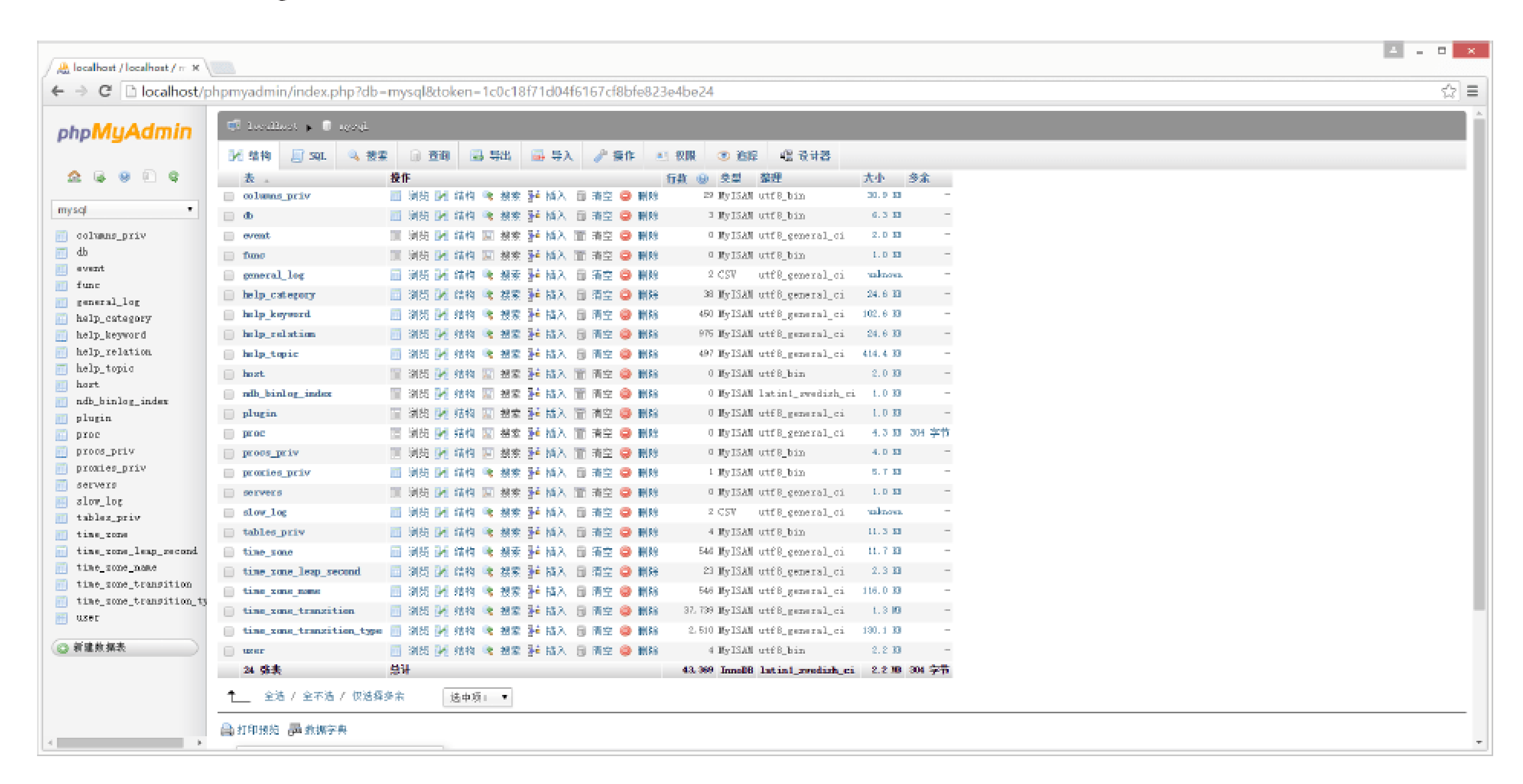


图17.2

在默认情况下，PHPMyAdmin有两个用户名，分别是pma和root。其中，root是管理员权限，而pma是普通用户权限，但是二者在默认状态下均无密码，所以我们点击admin按钮后就直接进入了phpMyAdmin的界面。另外，还可以在浏览器中直接输入http://localhost/phpmyadmin/地址进入。

数据库中存在这样可以不用密码就能登录的账户是不安全的，所以需要创建新的管理员账户并删除原有的root账户。在主界面点击“权限”选项卡，然后单击root账号后面的“编辑权限”链接，在弹出的对话框中有一个修改登录信息/复制用户的区域，如图17.3所示。



图17.3

在这个对话框中输入用户名和密码，其他参数保持不变，然后点击“执行”按钮，即可创建一个与root相同权限的用户。

再次编辑root账号，勾选“从用户表中删除旧用户”，其他参数不变，点击“执行”按钮后，即可删除root账户。



17.2 MySQL数据库基础

数据库可以看做是一个存储数据对象的容器，在MySQL中，这些数据对象包括以下几种。

（1）表：表是所有关系型数据中最重要的对象，是一种用于存储和操作数据的结构。

（2）视图：视图是将多个表中的字段组织在一起形成的一个虚拟表。视图本身并不存储数据，只定义基本的表结构。视图可以像表一样进行查询、修改、删除和更新的操作。

（3）索引：索引是为了提高表的检索效率，在数据表的一列或多列上创建的一种结构。这种结构根据索引表达式的值进行逻辑排序，它可以实现对数据的快速访问。

（4）约束：约束是保证数据库中数据唯一性与完整性的一种保证，最常用的约束就是主外键约束，主键约束当前表记录的唯一性，外键约束当前表记录与其他表的关系。

（5）存储过程：存储过程是一组能完成特定功能的SQL语句集合。这个语句集合经过编辑后存储在数据库中，存储过程具有输入、输出和输入/输出参数，它可以由程序、触发器或

另一个存储过程调用从而激活它，实现代码段中的SQL语句。

（6）触发器：触发器是一个被指定关联到一个表的数据库对象，触发器是不需要调用的，当一个表的特别事件出现时，它会被激活。触发器的代码是由SQL语句组成的，因此用在存储过程中的语句也可以用在触发器的定义中。触发器与表的关系密切，用于保护表中的数据。当有操作影响到触发器保护的数据时，触发器自动执行，例如，通过触发器实现多个表间数据的一致性。当对表执行INSERT、DELETE或UPDATE操作时，将激活触发器。

（7）存储函数：存储函数与存储过程类似，也是由SQL和过程式语句组成的代码片段，并且可以从应用程序和SQL中调用。但存储函数不能拥有输出参数，因为存储函数本身就是输出参数。存储函数必须包含一条RETURE 语句，从而返回一个结果。

（8）事件：事件与触发器类似，都是在某些事件发生时启动。不同的是触发器是在数据库上启动一条语句时被激活，而事件是在相应的时刻被激活。



17.3 MySQL表结构

MySQL是一种关系型数据库，表是数据库中一个非常重要的对象，下面我们简单介绍一些与表有关的基础知识。

（1）表结构：表结构由列名和数据类型组成。每个列名表示表中的一类数据，而数据类型指定了这列是哪一类数据。

（2）记录：表中的每一行称为一个记录。

（3）字段：每个记录由若干个数据项构成，构成记录的每个数据项称为字段。

（4）空值：空值null通常表示未知、不可用或在以后添加的数据。允许为空值的列在添加数据的时候可以不指定数据值，反之则必须指定一个数据值。

（5）主键：主键是用于区别表中所有数据的唯一标识。主键不能为空，且必须唯一。一个表中可以有一个列是主键，也可以是多个列共同组成主键。



17.4 MySQL数据类型

数据类型指定了表中的列应该使用哪种类型的数据，MySQL在创建表的时候，要求必须为其指定数据类型。MySQL提供了丰富的数据类型，如表17.1所示。

表17.1

属性	描述
整数型	BINGINT，INT，SMALLINT，MEDIUMINT，TINYINT
精确数据值	DECIMAL，NUMBERIC
浮点型	FLOAT，REAL，DOUBLE
位型	BIT
字符型	CHAR，VARCHAR，LONGVARCHAR，LONGTEXT
Unicode字符型	NCHAR，NVARCHAR
BLOB类型	TINYBLOB，BLOB，MEDIUMBLOB，LONGBLOB
文本型	TEXT，TINYTEXT
二进制型	BINARY，VARBINARY
日期时间类型	DATE，TIME，DATETIME，TIMESTAMP，YEAR



17.5 创建数据库和表

了解了MySQL数据库的相关知识后，下面我们介绍如何使用phpMyAdmin对数据库和表进行相关操作。

17.5.1 创建数据库

登录phpMyAdmin界面后，点击“数据库”标签页，在新建数据库的第一个文本框中输入数据库的名称，然后在后面的下拉列表中选择数据库对应的字符集，最后点击“创建”按钮，这样就创建了一个新的数据库，如图17.4所示。新创建的数据会出现在左边的数据库列表中，如图17.5所示。

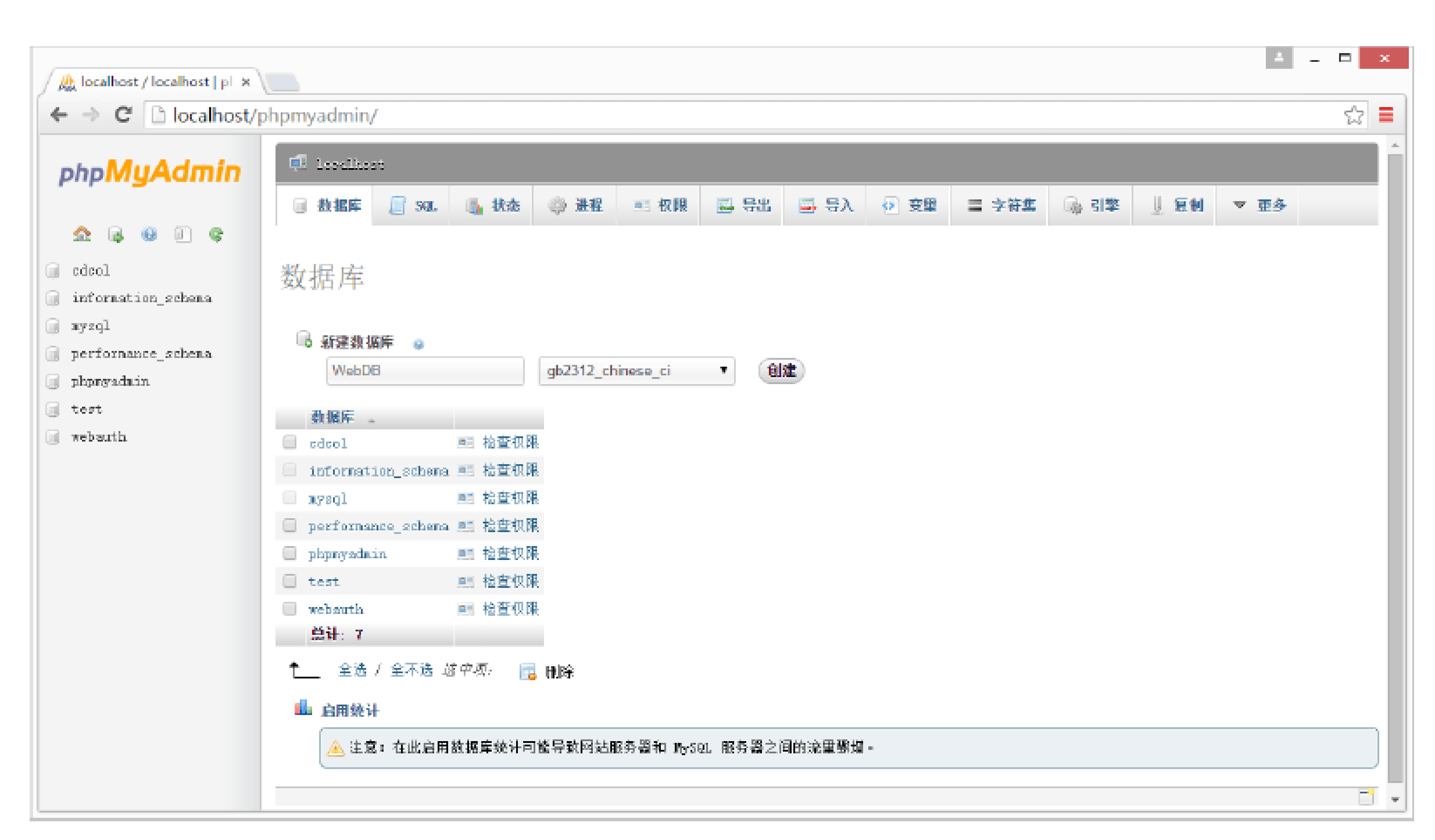


图17.4

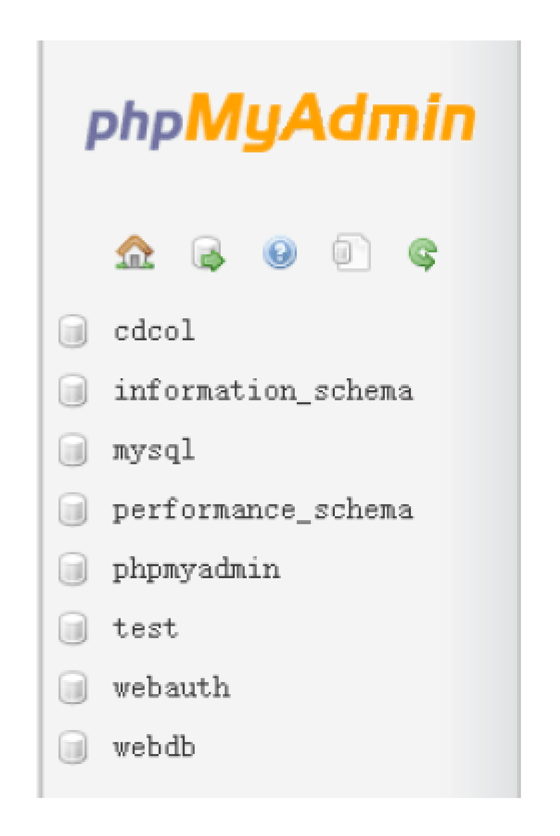


图17.5

17.5.2 指定数据库用户

在如图17.5所示图形中选中新创建的数据库，进入数据库管理界面，如图17.6所示。

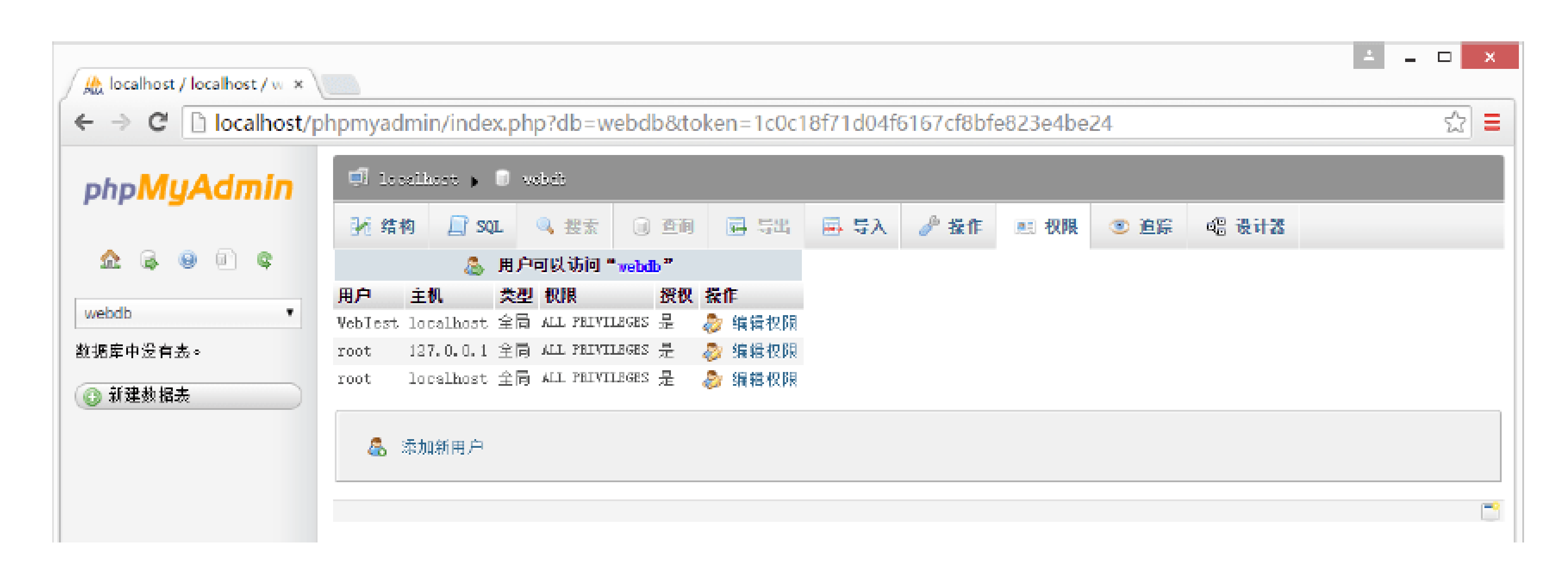


图17.6

点击“权限”标签，然后点击“添加新用户”链接，打开创建新用户界面，如图17.7所示。在这个界面中填写数据库新用户的名称，选择主机并设置密码，点击“执行”按钮后为新建的数据库创建一个用户。



图17.7

17.5.3 创建数据表

选中新建的数据库进入数据库管理界面，然后单击“创建数据表”按钮，填写数据表名称、字段名称、字段类型等信息，如果需要添加更多的字段，可以单击“执行”按钮，如图17.8所示。添加完所有字段后，单击“保存”按钮完成创建数据表操作。新添加的数据表会出现在左侧列表中，单击新建的数据表，可以看到新建数据表的字段等信息，如图17.9所示。

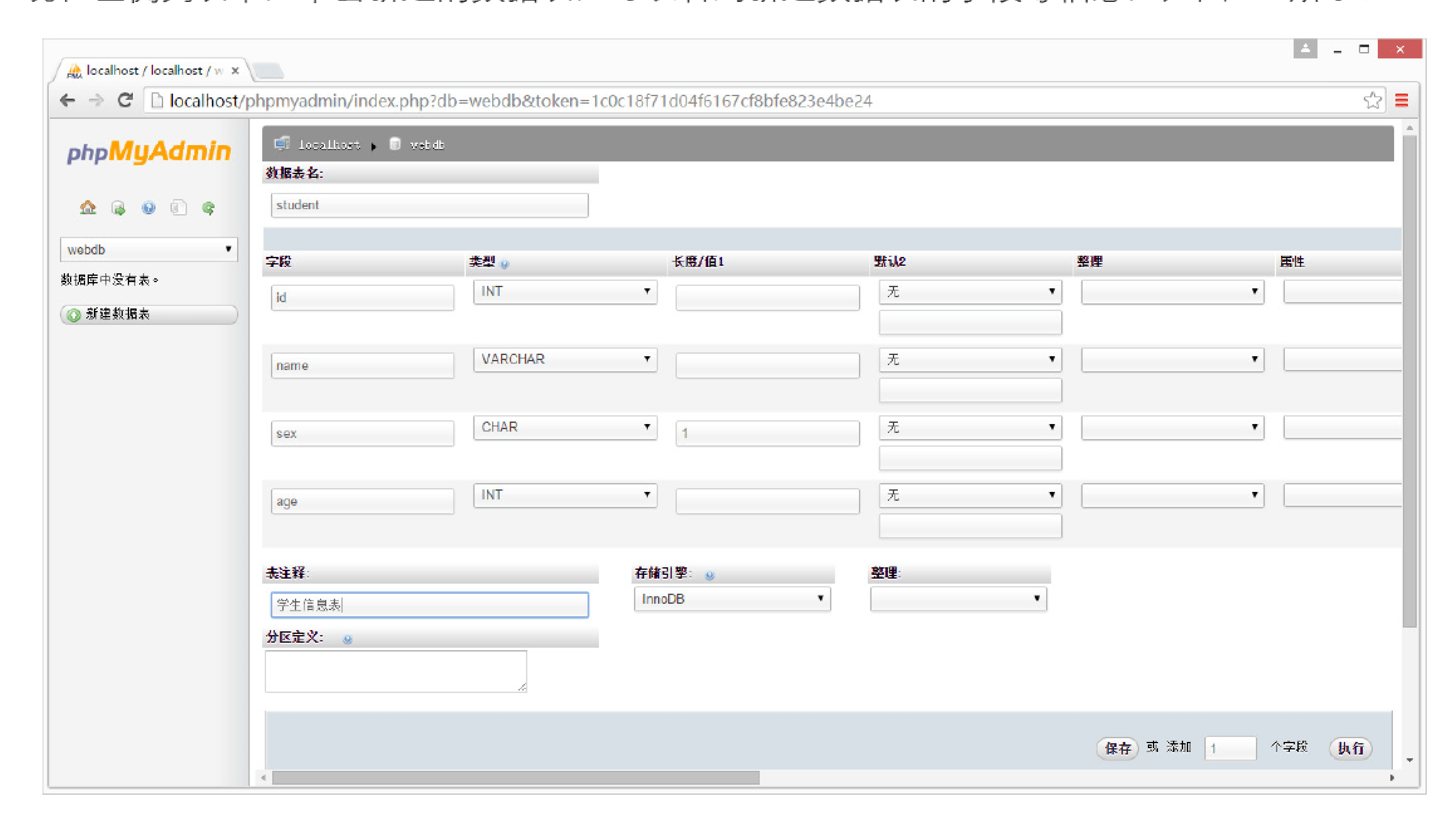


图17.8

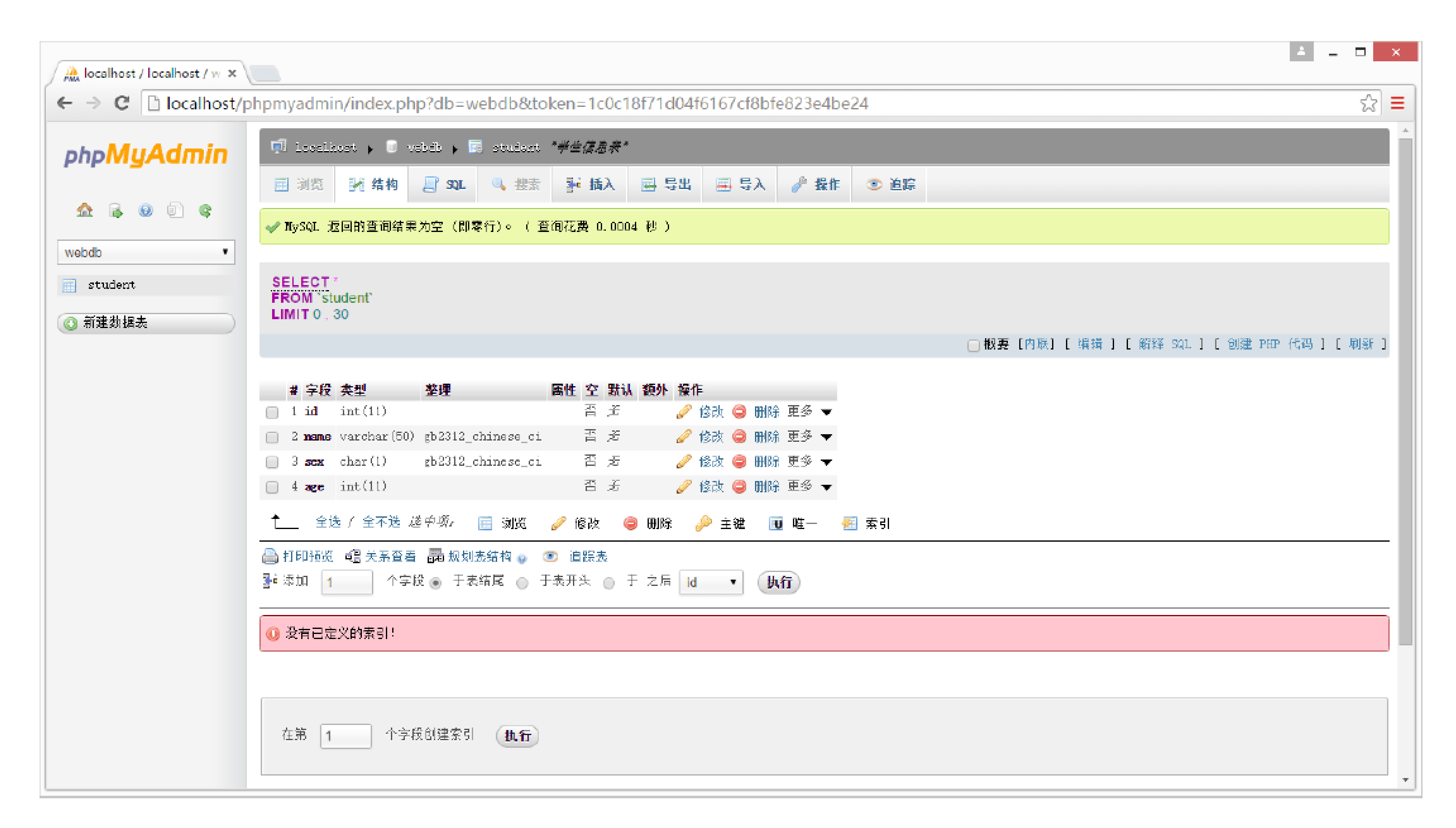


图17.9

17.6 添加、修改、删除和查询数据

创建数据表后，我们就可以在数据表中对数据进行相关操作，包括添加、修改、删除和查询数据的操作。

17.6.1 添加数据

选中创建的数据表后，单击插入选项，打开如图17.10所示的录入窗口，填写相关信息后，单击“执行”按钮，即可添加数据。

17.6.2 修改数据

选中创建的数据表，可以查看当前数据表中的数据，如图17.11所示。单击对应数据行前面的“编辑”按钮，可以重新打开如图17.10所示的界面，对选中的数据进行修改后，单击“执行”按钮即可修改选中的数据。

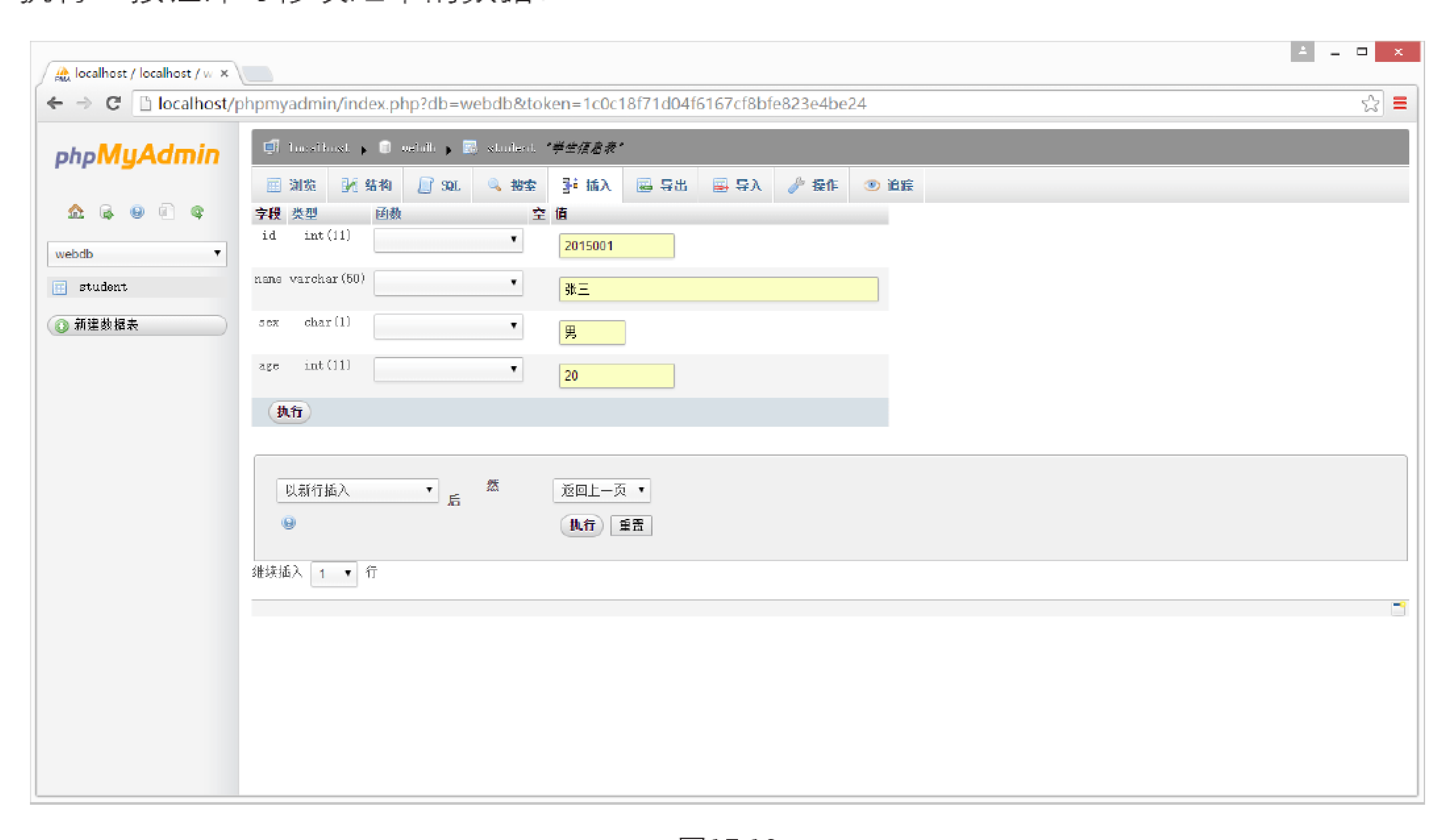


图17.10

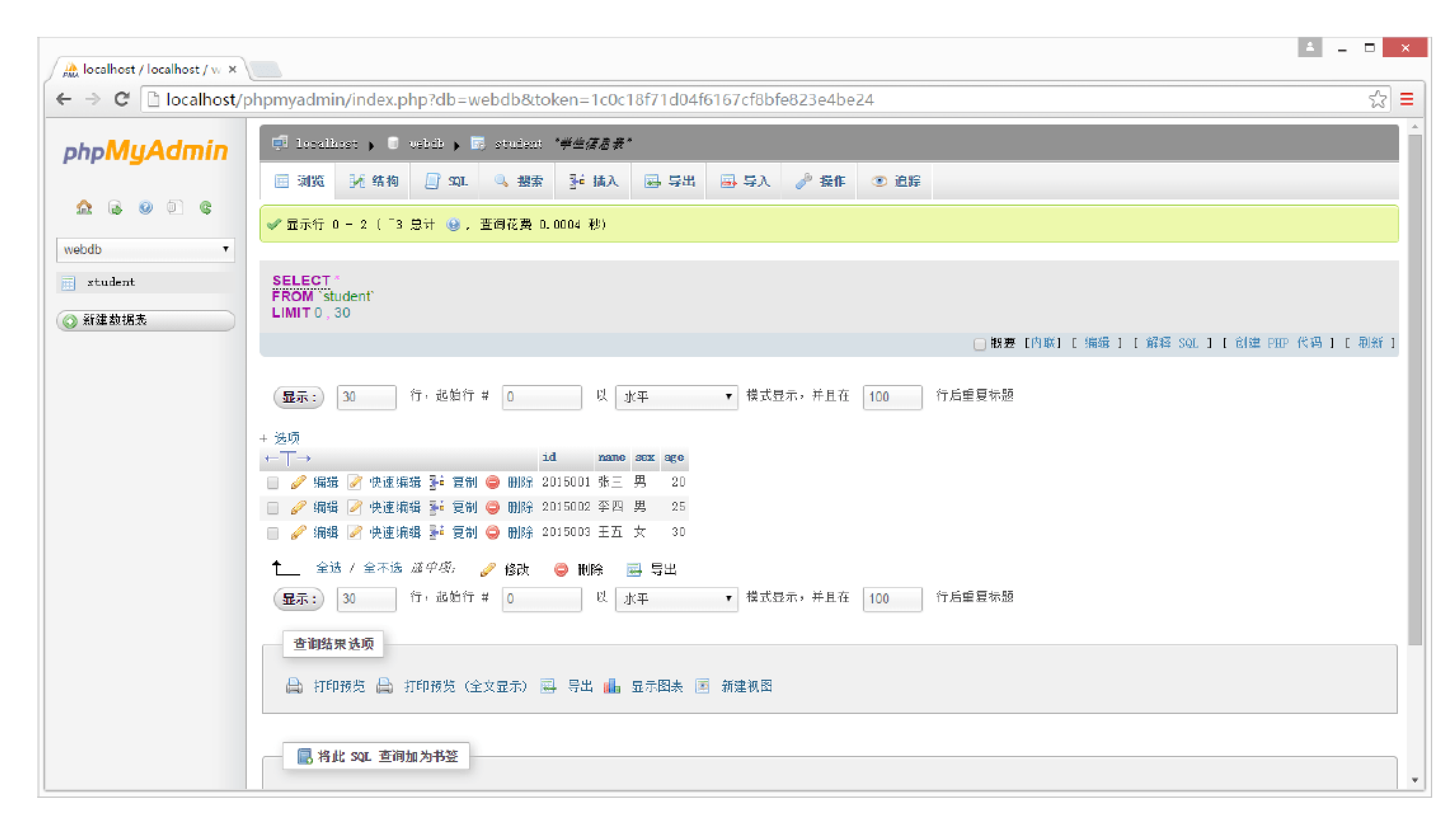


图17.11

17.6.3 删除数据

在如图17.11所示的图形中选中要删除的数据，单击“删除”按钮，弹出如图17.12所示的提示框，单击“确定”按钮后即可删除选中的数据。

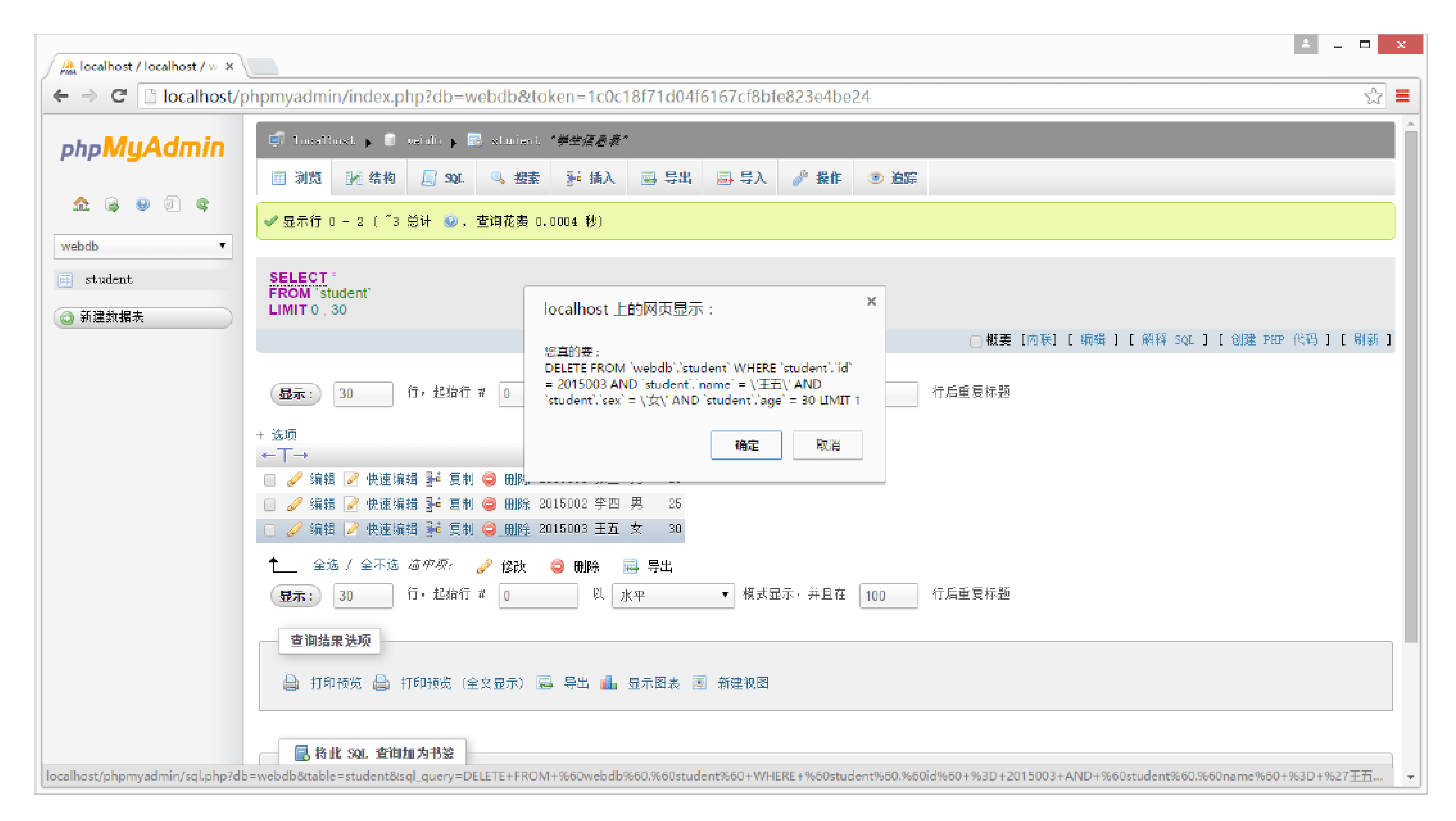


图17.12

17.6.4 查询数据

使用phpMyAdmin界面执行查询操作的方法非常简单，单击“数据表”即可完成查询数据的操作，如图17.13所示。用户可以在“显示”文本框中输入要查询数据的数目来控制返回结果的数量，还可以控制从多少行开始查询，以水平或垂直的方式显示结果等。

可视化查询界面通过鼠标操作完成相关查询的功能比较有限，而且操作不是很方便，要想高效地执行复杂查询操作，还是需要执行相关的SQL查询。



图17.13

第18章 使用Dreamweaver创建PHP+MySQL动态网站

Dreamweaver是一款非常流行的用于创建可视化网页的工具，它不仅可以创建可视化页面效果，还可以配合PHP和MySQL创建动态网站。本章将通过具体案例，详细介绍如何使用Dreamweaver创建PHP+MySQL的动态网站。



18.1 Dreamweaver与PHP的整合

在前面的课程中，我们介绍了如何在Dreamweaver中编写PHP代码，要创建一个网站，就需要创建很多的页面，使用Dreamweaver可以对这些页面进行很好的管理。在Dreamweaver中可以通过以下步骤创建站点并开发PHP程序。

01 启动Dreamweaver软件，选择“站点”→“新建站点”命令，打开“站点设置对象LibrarySys”对话框，如图18.1所示。在该对话框中的“站点名称”文本框中输入网站站点的名称，并指定本地站点文件夹为xampp下的程序发布目录。

图18.1

02 切换到左边列表中的“服务器”选项，然后单击右边的“加号”按钮，在打开的对话框中设置服务器名称为“LibrarySys”，连接方式为“本地/网络”，服务器文件夹选择xampp目录下的LibrarySys文件夹，WebURL设置为http://localhost/ LibrarySys/，效果如图18.2所示。

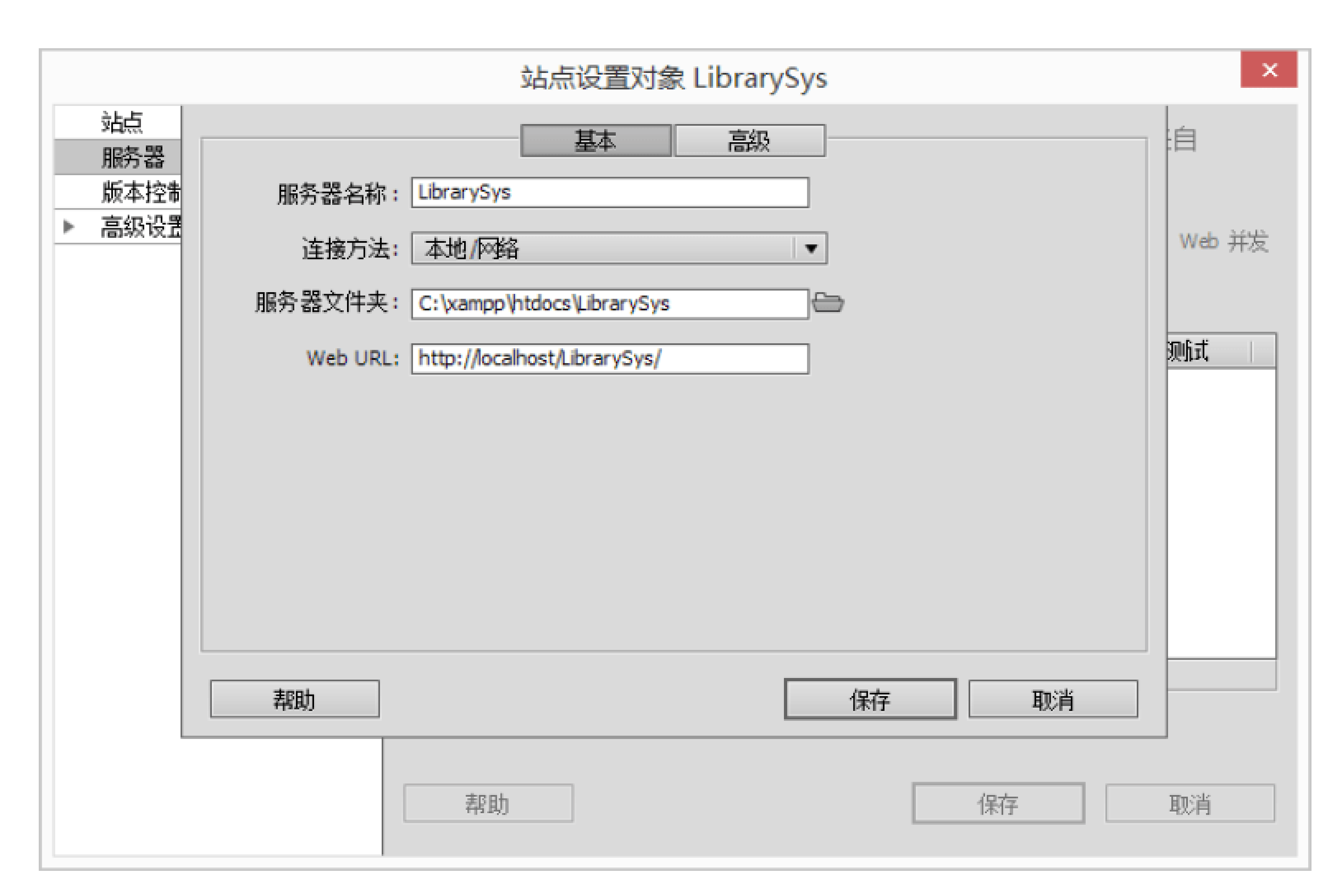


图18.2

03 切换到左边列表中的“高级设置”选项，然后选择“本地信息”选项，在右边的区域中设置“默认图像文件夹”地址，然后选择链接相对于“站点根目录”，效果如图18.3所示。

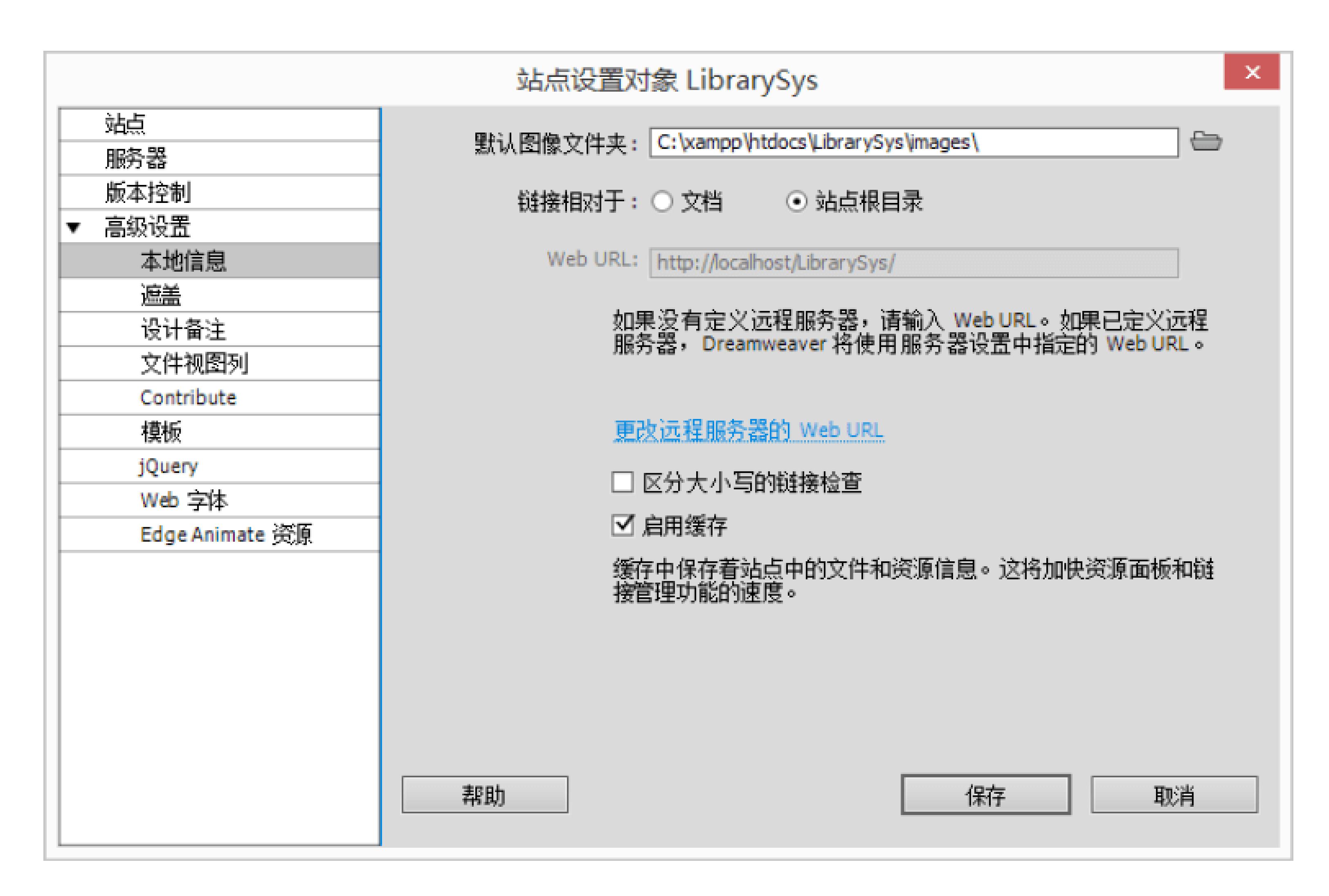


图18.3

04 创建后的站点效果如图18.4所示。

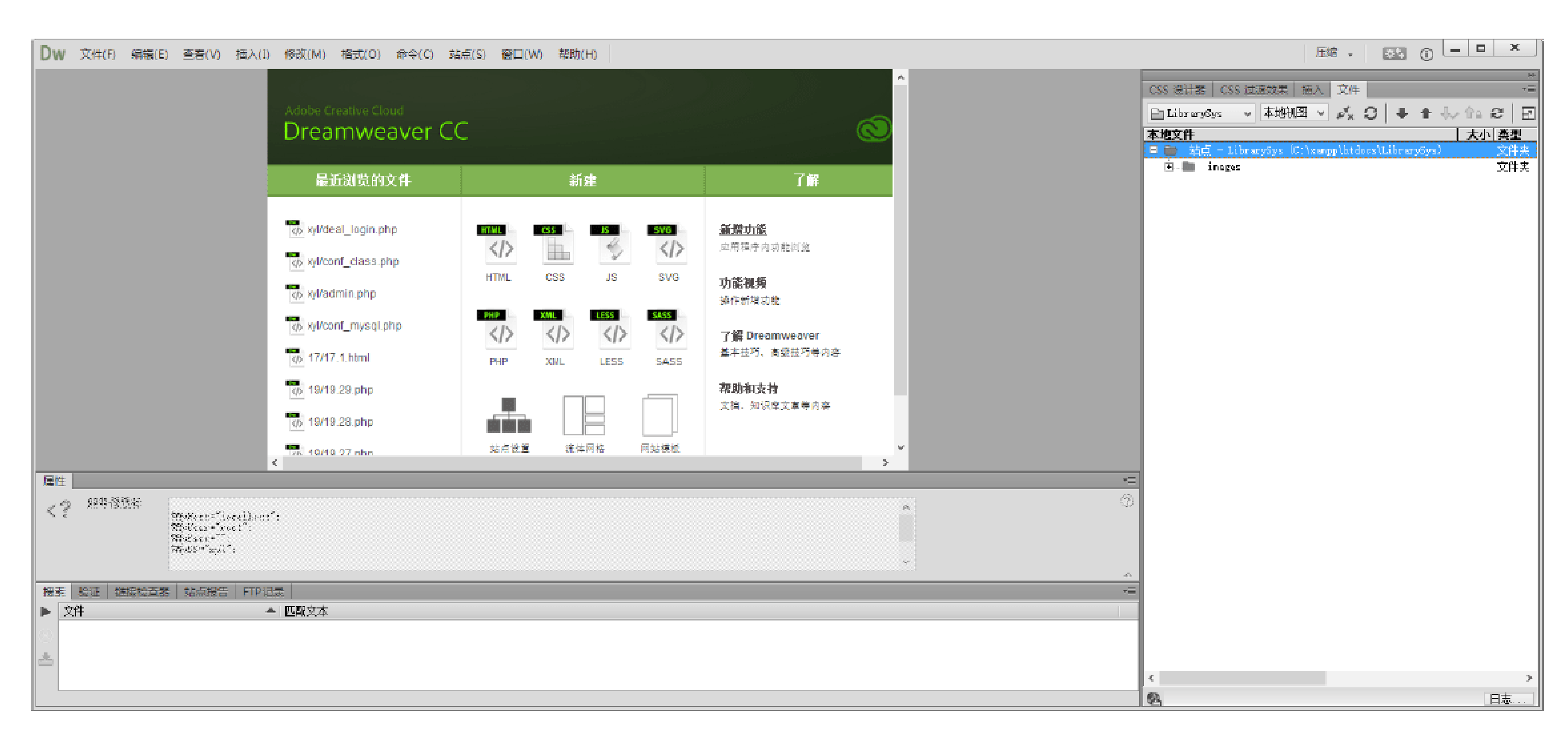


图18.4

05 在Dreamweaver中创建一个名为index.php的PHP文件，并在页面中输出一段文件，将其保存在创建站点的根目录下，效果如图18.5所示。

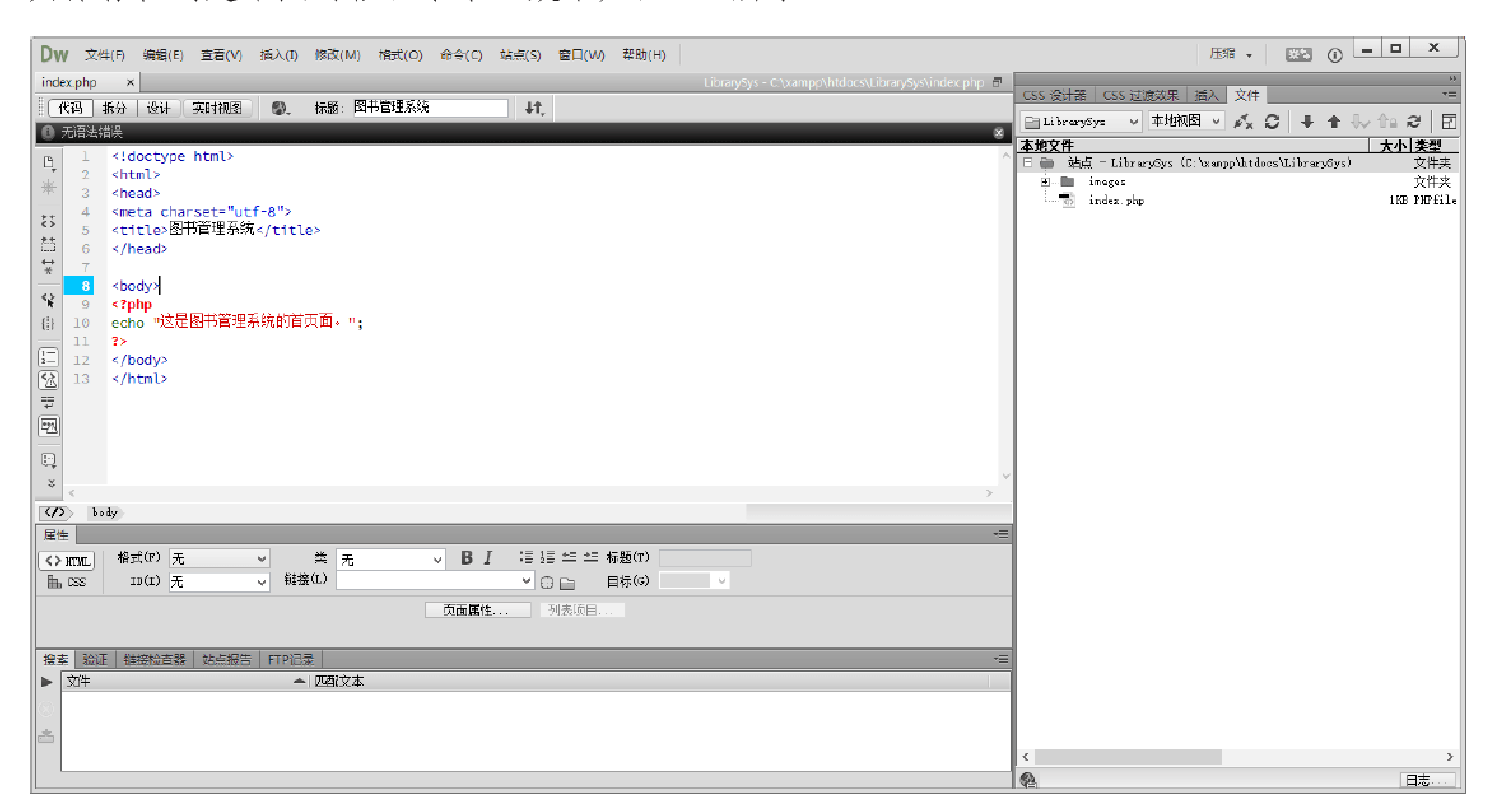


图18.5

06 按F12功能键，在浏览器中浏览创建的页面效果，如果看到如图18.6所示的页面，则说明使用Dreamweaver成功创建了一个PHP站点。

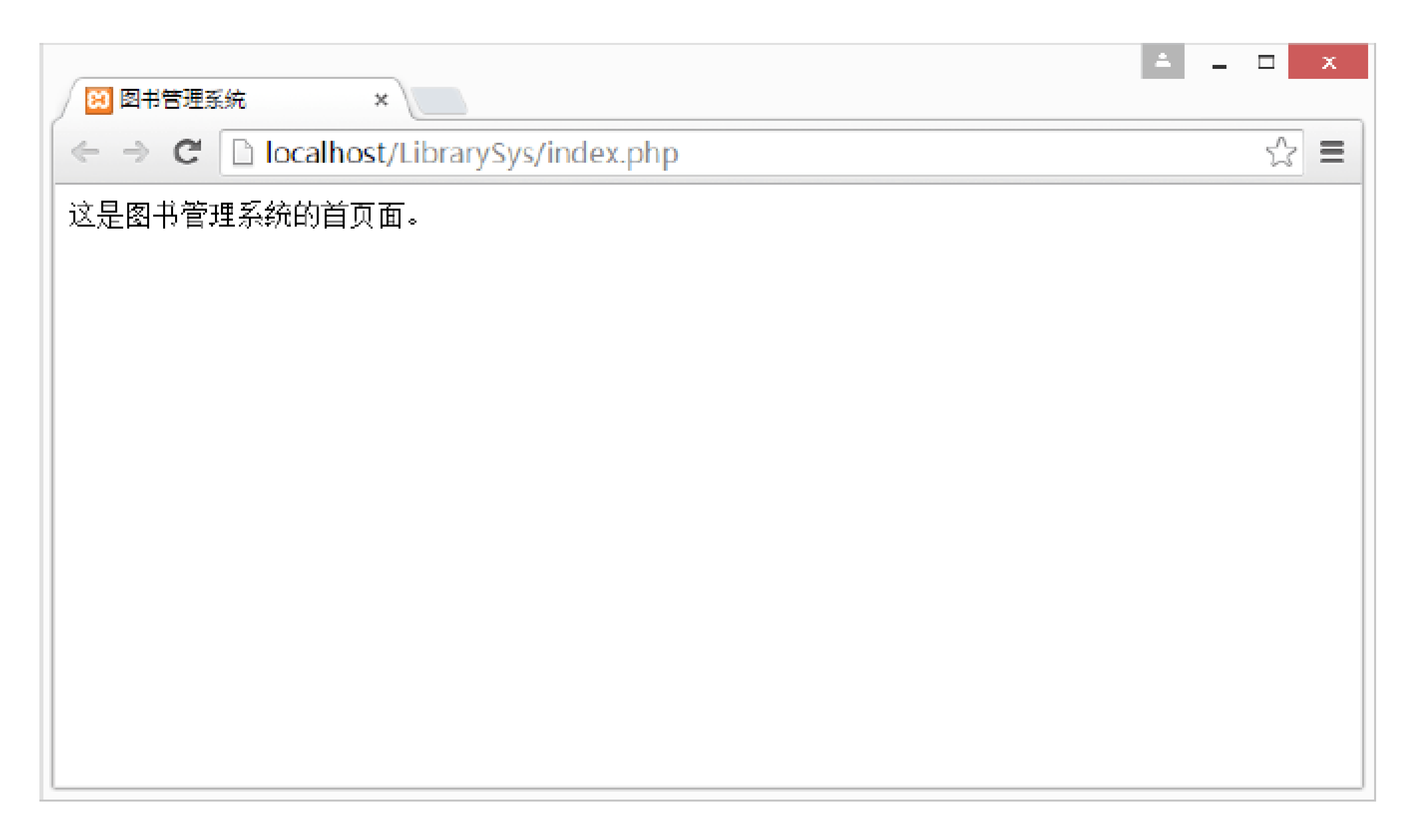


图18.6



18.2 创建会员管理动态网站

为了说明使用Dreamweaver创建PHP+MySQL动态网站的过程，下面以图书管理系统动态网站为例，详细介绍创建这个网站的整个过程。

18.2.1 总体规划

图书管理系统是学习各种编程语言中一个比较经典的案例，在本例中，我们为这个系统规划了以下几个功能。

（1）借阅排行：根据当前系统记录的图书借阅情况，查询图书的借阅信息，并根据借阅次数对图书进行排序。

（2）借阅管理：该功能主要实现对图书借阅的管理操作，包括图书借阅、图书续借和图书归还3个子功能。图书借阅功能主要实现新增图书借阅信息，图书续借功能主要实现续借图书功能，图书归还功能主要实现归还图书功能。

（3）读者管理：所有借阅图书的读者都必须在系统中进行登记，这个功能主要实现对读者类型的管理和读者信息的管理。不同类型的读者可借阅图书的数量不同。

（4）图书管理：该功能主要实现系统中图书信息的管理，包括新增图书信息和修改图书信息功能。

（5）系统设置：系统设置是该系统中设置各项参数的地方，包括设置图书类型、出版社、权限、用户和借阅量等。

（6）修改密码：该功能主要用于修改当前登录用户的密码。

（7）退出：退出系统。

18.2.2 数据字典

数据字典是指系统涉及的各表的名称、字段类型和相关说明信息，根据我们的规划，图书管理系统中用到的表信息如表18.1~表18.7所示。

表18.1 系统用户表tb_sysuser

字段名称	字段类型	说明
id	int(10)	主键
name	varchar(30)	用户名
pwd	varchar(30)	密码

表18.2 读者类型表tb_readertype

字段名称	字段类型	说明
id	int(10)	主键
name	varchar(50)	读者类型名称
number	Int(4)	可借阅量

表18.3 读者信息表tb_reader

字段名称	字段类型	说明
id	int(10)	主键
name	varchar(30)	名称
sex	varchar(4)	性别
barcode	varchar(30)	条形码
vocation	varchar(50)	职业
birthday	date	出生日期
papertype	varchar(10)	证件类型
papernum	varchar(20)	证件编号
tel	varchar(20)	电话
email	varchar(100)	电子邮件
createdate	timestamp	创建时间
opearator	varchar(30)	操作员
remark	mediumtext	备注
typeid	int(10)	读者类型id

表18.4 出版社信息表tb_publishing

字段名称	字段类型	说明
id	int(10)	主键
isbn	varchar(20)	ISBN
pubname	varchar(30)	出版社名称

表18.5 图书类型表tb_booktype

字段名称	字段类型	说明
id	int(10)	主键
typename	varchar(30)	图书类型名称

表18.6 图书信息表tb_bookinfo

字段名称	字段类型	说明
id	int(10)	主键
barcode	varchar(30)	条形码
bookname	varchar(70)	书名
booktypeid	Int(10)	图书类型编码
author	varchar(30)	作者
isbn	varchar(20)	isbn编码
price	float(8,2)	价格
storage	int(10)	馆藏量
intime	Date	收藏时间
operator	varchar(30)	操作员
del	tinyint(1)	是否删除

表18.7 tb_borrow

字段名称	字段类型	说明
id	int(10)	主键
readerid	int(10)	借阅人id
bookid	int(10)	借阅图书id
borrowtime	timestamp	借阅时间
backtime	timestamp	归还时间
operator	varchar(30)	操作人
ifback	tinyint(1)	是否归还

启动xampp，登录phpMyAdmin管理界面，创建一个名为WebDB的数据库，然后在这个数据库中创建以上所有表结构，并在部分表中输入基础数据。

18.2.3 登录页面实现

登录页面的实现首先需要编写HTML页面代码，实现登录界面，然后创建数据库连接，最后实现用户登录验证，下面进行具体介绍。

1. 编写HTML页面代码

登录页面是整个系统的入口，先使用Dreamweaver在创建的站点下创建一个名为login.php的登录页面，页面相关HTML代码如下所示：

```
<!doctype html>
```

```
<html>
<head>
<meta charset="utf-8">
<title>图书管理系统</title>
<style type="text/css">
* {
margin: 0;
padding: 0;
}
body {
background-color: #66A2EB;
}
#login {
width: 400px;
height: 230px;
margin: 50px auto;
font-weight: bold;
text-align: center;
background-color: #336594;
border-radius: 10px;
color:white;
}
#login ul li {
list-style: none;
margin: 5px auto;
}
#login ul li:first-child{
margin:30px auto;
font-size:24px;
color:#E47E24;
}
#btnSubmit, #btnReset {
width: 75px;
height: 24px;
font-weight: bold;
font-size: 18px;
margin: 20px 0 0 0;
color:white;
background-color:#35749C;
}
#errorMessage {
font-size: 10px;
color: red;
}
</style>
<script language="javascript">
function check(form){
if (form.adminName.value==""){
document.getElementById("errorMessage").innerHTML="请输入管理员名称!";
form.adminName.focus();
return false;
```

```
    }
    if (form.adminPass.value==""){
    document.getElementById("errorMessage").innerHTML="请输入管理员密码!";
    form.adminPass.focus();
    return false;
    }
    }
    </script>
    </head>
    <body>
    <fieldset id="login">
       <form id="loginForm" name="loginForm" action="/LibrarySys/index.php"
method="post">
         <ul>
           <li id="title">
             <span>图书管理系统</span>
           </li>
           <li><span>管理员名称：</span>
             <input type="text" id="adminName" name="adminName"/>
           </li>
           <li><span>管理员密码：</span>
             <input type="password" id="adminPass" name="adminPass"/>
           </li>
           <li>
                 <input type="submit" id="btnSubmit" name="btnSubmit"
onClick="return check(loginForm);" value="登录"/>
             <input type="reset" id="btnReset" name="btnReset" value="重置"/>
           </li>
           <li>
             <label id="errorMessage" name="errorMessage" ></label>
           </li>
         </ul>
       </form>
    </fieldset>
    </body>
    </html>
```

这个页面的代码比较简单，这里就不再赘述。需要注意的是，在点击登录按钮的时候做了一个用户名和密码的JavaScript非空验证，如果某一项输入为空，则在页面下面的label标签中显示提示信息，效果如图18.7所示。如果用户名和密码都输入，那么点击登录按钮后直接进入index.php页面。



图18.7

在没有做动态验证之前，点击登录按钮直接进入index.php页面，这是因为我们在login.php页面中的form表单中指定它的action跳转地址为index.php页面。下面我们要链接MySQL数据库，对输入的用户名和密码进行数据库验证。

2. 创建数据库连接

创建一个conn.php的页面，该页面用于设置php链接MySQL数据的连接语句，详细代码如下所示：

```
<?php
      $conn=mysql_connect("localhost","root","") or die("数据库服务器连接错误".mysql_error());
     mysql_select_db("webDB",$conn) or die("数据库访问错误".mysql_error());
     mysql_query("set names gb2312");
?>
```

3. 用户登录验证

创建一个checklogin.php页面，并将form表单的action地址指向该页面，点击登录按钮之后，该页面将接收login.php页面提交的用户名和密码，然后链接MySQL数据库，从数据库中查询对应的数据。如果查询到结果，则表示登录成功，否则表示登录失败。该页面详细代码如下所示，注意要在第一行指定页面编码，否则出现错误时，错误信息将会显示乱码。

```
<meta charset="utf-8">
<?php
session_start();
$adminName=$_POST['adminName'];
$adminPass=$_POST['adminPass'];
class Login{
var $username;
var $userpass;
function Login($name,$pass){
$this->username=$name;
$this->userpass=$pass;
}
function checkLoagin(){
```

```
    include("conn.php");
    $sql=mysql_query("select * from tb_sysuser where name='".$this->username."' and pwd='".$this->userpass."'",$conn);
    $info=mysql_fetch_array($sql);
    if($info==false){
    echo "<script language='javascript'>alert('用户或密码错误，请重新输入！');history.back();</script>";
    return false;
    }else{
    echo "<script language='javascript'>alert('登录成功！');window.location='index.php'</script>";
    $_SESSION['admin_name']=$info['name'];
    $_SESSION['admin_pwd']=$info['pwd'];
    return true;
    }
    }
    }
    $obj=new Login(trim($adminName),trim($adminPass));
    $obj->checkLoagin();
    ?>
```

重新打开登录界面，输入错误的用户名和密码，然后点击登录按钮，此时将显示登录失败，效果如图18.8所示。再次输入正确的用户名和密码，点击登录按钮，此时显示登录成功，并跳转到index.php页面，效果如图18.9所示。



图18.8



图18.9

18.2.4 系统主界面实现

成功实现登录页面后，下面我们来设计系统的主界面。我们将主界面分为3个部分，最上面的一部分为页面的头部，中间部分是主要内容，页面的最底部是版权和其他信息。

1. 添加logo和信息

页面头部包含网站的logo、名称、当前系统时间、当前登录用户名以及导航条，这些内容将是所有页面共享的内容，所以我们单独为这个部分创建一个naviagtion.php页面，这个页面中logo、页面名称、当前系统时间和登录用户名的HTML代码如下所示：

```
<style type="text/css">
header {
background-color: #4C94C1;
font-weight:bold;
}
header div {
margin-bottom:1px;
}
header div img{
width:80px;
height:80px;
margin:20px 20px 0px 50px;
}
header div h1{
display:inline-block;
color:#D307EE;
font-family:"华文彩云";
font-size:50px;
}
header div #loginUser {
text-align:right;
margin-right:30px;
padding-bottom:5px;
```

```
padding-top:5px;
color:white;
}
</style>
<?php session_start(); ?>
<header>
<div> <img src="/LibrarySys/images/logo01.jpg" />
<h1>图书管理系统</h1>
<div id="loginUser">
<span id="labtime"></span>
<script type=text/javascript>
setInterval("labtime.innerText=new Date().toLocaleString()",1000)
</script>
<span>当前登录用户：<?php echo $_SESSION['admin_name']; ?></span> </div>
</div>
</header>
```

在这段代码中，我们没有看到html、head等标签，这是因为这个页面将会在其他页面中引入，如果添加这些标签，就会出现标签重复的问题。获取当前系统时间的方法比较简单，我们只需要设置一个计时器，每秒钟给标签赋值当前系统时间即可，关键代码如下所示：

```
setInterval("labtime.innerText=new Date().toLocaleString()",1000)
```

在登录验证的时候，我们已经将当前登录的用户名和密码存储在session中，所以在这个页面的开始使用了以下代码调用session。

```
<?php session_start(); ?>
```

然后在需要输出用户名的地方再次使用php从session中获取到当前的用户名。编写完以上代码后，需要在index.php页面中引入这个文件，index.php页面的代码如下所示：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>图书管理系统</title>
<style type="text/css">
* {
margin: 0;
padding: 0;
}
body {
width: 800px;
margin: 0 auto;
}
</style>
</head>
<body>
<?php
include("navigation.php");
?>
```

```
</body>
</html>
```

这里的include就是将指定的页面引入当前页面，运行这段代码，登录成功后的效果如图18.10所示。

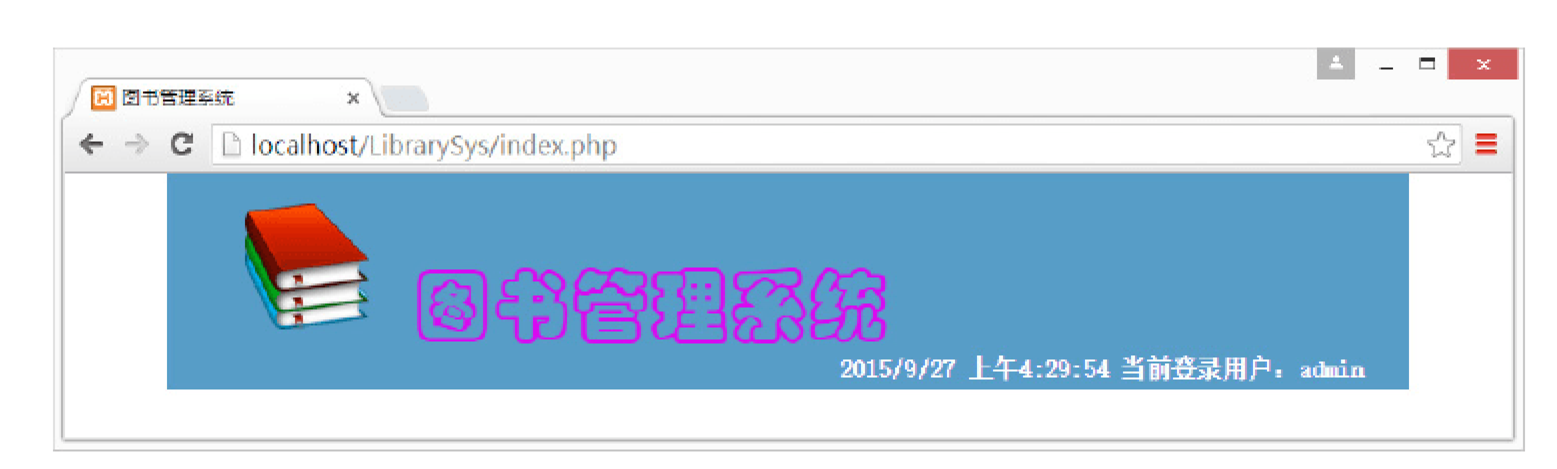


图18.10

2. 添加导航

页面的头部同样也包含一个导航菜单，我们将其设置为二级菜单，因为导航菜单对应的页面还没有创建，所以暂时先将各个链接的hred属性设置为#，相关代码如下所示：

```
<style type="text/css">
nav ul li {
list-style: none;
float: left;
position: relative;
margin-right: 1px;
}
nav ul li a {
text-decoration: none;
display: block;
padding: 5px;
color: white;
background-color: #4E85EA;
font-size:9px;
}
nav ul li:hover a {
background-color: #374256;
}
nav ul li ul {
display: none;
}
nav ul li:hover ul {
display: block;
position: absolute;
}
nav ul li:hover ul li {
margin-top: 1px;
}
nav ul li:hover ul li a {
display: block;
background-color: #374256;
width: 100px;
}
```

```
nav ul li:hover ul li a:hover {
color: white;
background-color: #316893;
}
</style>
<header>
<nav>
<ul>
<li><a href="#">首页</a></li>
<li><a href="#">借阅管理</a>
<ul>
<li><a href="#">图书借阅</a></li>
<li><a href="#">图书续借</a></li>
<li><a href="#">图书归还</a></li>
</ul>
</li>
<li><a href="#">读者管理</a>
<ul>
<li><a href="#">读者类型管理</a></li>
<li><a href="#">读者档案管理</a></li>
</ul>
</li>
<li><a href="#">图书管理</a></li>
<li><a href="#">系统设置</a>
<ul>
<li><a href="#">图书类型</a></li>
<li><a href="#">出版社</a></li>
<li><a href="#">权限</a></li>
<li><a href="#">用户</a></li>
</ul>
</li>
<li><a href="#">更改密码</a></li>
<li><a href="#">退出</a></li>
</ul>
</nav>
</header>
```

运行这段代码后，将鼠标停留在导航菜单上，效果如图18.11所示。



图18.11

3. 添加主体内容

首页中的主体内容将显示当前系统图书借阅排行榜，我们使用表格来显示这些数据。首先编写一个静态页面，相关代码如下所示：

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>图书管理系统</title>
<style type="text/css">
section {
clear: both;
padding-top:1px;
}
section img{
width:25px;
height:25px;
}
section span{
font-weight:bold;
margin-left:10px;
}
section table tr {
font-size: 9px;
}
section table tr:first-child {
text-align: center;
background-color: #B72729;
font-weight: bold;
color: white;
}
section table tr:nth-child(2n) {
background-color:#B4D6E0;
}
</style>
</head>
<body>
<section>
<img src="/LibrarySys/images/bookorder.png" /><span>图书借阅排行榜</span>
<table cellspacing="0">
<tr id="theader">
<td width="50px" height="25">排名</td>
<td width="120px">图书条形码</td>
<td width="150px">图书名称</td>
<td width="100px">图书类型</td>
<td width="150px">出版社</td>
<td width="150px">作者</td>
<td width="75px">定价(元)</td>
<td width="75px">借阅次数</td>
</tr>
<?php
$i=1;
do{
?>
```

```
    <tr>
    <td height="25" align="center">排名</td>
    <td style="padding:5px;">图书条形码</td>
    <td style="padding:5px;">图书名称</td>
    <td style="padding:5px;">图书类型</td>
    <td align="center">出版社</td>
    <td align="center">作者</td>
    <td align="center">定价（元）</td>
    <td align="center">借阅次数</td>
    </tr>
    <?php
    $i=$i+1;
    }while($i<10)
    ?>
    </table>
    </section>
    </body>
    </html>
```

在这段代码中，表头只显示了一行，而数据行则通过一个循环，显示了一些静态数据。运行这段代码后，效果如图18.12所示。



图18.12

下面需要通过查询从数据库中获取这些数据，在页面的开始部分引入数据库连接，代码如下：

```
    include("conn.php");
```

然后在do…while的前面通过php查询数据库中图书的信息，并返回结果，相关代码如下：

```
    <?php
    $sql=mysql_query("select * from (select bookid,count(bookid) as degree from tb_borrow group by bookid) as borr join (select b.*,c.name as bookcasename,p.pubname,t.typename from tb_bookinfo b left join tb_bookcase c on b.bookcase=c.id join tb_publishing p on b.ISBN=p.ISBN join tb_booktype t on b.typeid=t.id where b.del=0) as book on borr.bookid=book.id order by borr.degree desc limit 10");
```

```
$info=mysql_fetch_array($sql);
$i=1;
do{
?>
```

同时还要修改while的条件，代码如下：

```
<?php
$i=$i+1;
}while($info=mysql_fetch_array($sql));
?>
```

运行这段代码后，效果如图18.13所示。



图18.13

4. 添加底部信息

网页的底部信息是一个静态页面，创建一个名为footer.php的页面，在该页面中编写以下代码：

```
<meta charset="gb2312">
<style type="text/css">
footer{
margin-top:5px;
}
footer p{
text-align:center;
font-size:9px;
}
</style>
<footer>
<hr>
<p>CopyRight &copy; 2015 ***图书馆<br>
Tel:010-22225555   Email:***@***.com</p>
</div>
</footer>
```

在index.php页面中使用以下代码引入footer.php页面。

```
<?php
include("footer.php");
?>
```

运行这段代码后，效果如图18.14所示。



图18.14

18.2.5 读者管理

读者管理模块主要包括读者类型管理和读者档案管理两个部分，这两个部分都实现了基础信息的展示功能。

1. 读者类型管理

新建一个名为readertype.php的页面，在该页面中编写以下代码：

```
<!doctype html>
<html>
<head>
<meta charset="gb2312">
<title>图书管理系统</title>
<link type="text/css" href="/LibrarySys/css/style.css" rel="stylesheet"/>
</head>
<body>
<?php
include("navigation.php");
include("conn.php");
?>
<section> <img src="/LibrarySys/images/bookorder.png" /><span>读者类型</
span> <a id="addreaderlink" href="/LibrarySys/readertypeedit.php" >添加读者类型
信息</a>
    <table cellspacing="0">
      <tr id="theader">
        <td width="400px" height="25">读者类型名称</td>
        <td width="200px">借阅数量</td>
        <td width="200px">编辑 删除</td>
```

```
        </tr>
        <?php
    $sql=mysql_query("SELECT * FROM tb_readertype");
    $info=mysql_fetch_array($sql);
    $i=1;
    do{
    ?>
        <tr>
          <td height="25" align="center"><?php echo $info[ ‘name’ ] ?></td>
          <td align="center" ><?php echo $info[ ‘number’ ] ?></td>
            <td align="center" ><a href="readertypeedit.php?id=<?php echo
$info['id'] ?>">编辑</a> <a href="readertypedel.php?id=<?php echo
$info['id'] ?>">删除</a></td>
        </tr>
        <?php
    $i=$i+1;
    }while($info=mysql_fetch_array($sql));
    if($i<10){
    for($i;$i<=20;$i++){
    ?>
       <tr>
          <td height="25" align="center"></td>
          <td align="center" ></td>
          <td align="center" ></td>
        </tr>
        <?php
    }
    }
    ?>
      </table>
    </section>
    <?php
    include("footer.php")
    ?>
    </body>
    </html>
```

在这段代码中，页面的主体风格与主界面类似，右上角的“添加读者信息”超链接对应readertypeadd.php页面，用于添加新的读者类型；每行数据的后面都有一个“编辑”和“删除”超链接，分别对应readertypeedit.php和readertypedel.php页面，用于对当前数据进行编辑操作；另外还有一个操作页面readertypeupdate.php，用于执行更改操作并显示操作结果。读者类型管理主界面效果如图18.15所示。

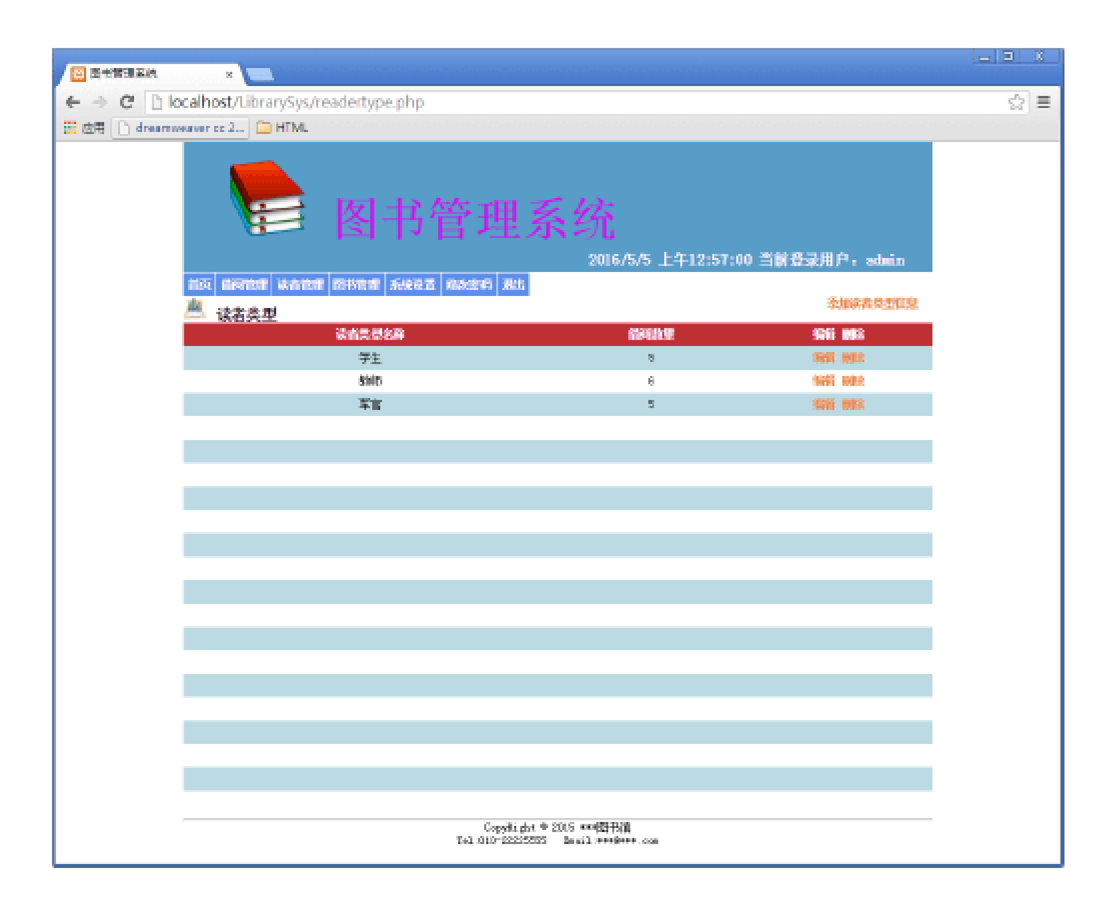


图18.15

其他几个php页面的代码如下所示。

```
readertypeadd.php
<meta charset="gb2312">
<?php
include("conn.php");
$name=$_POST[ ‘readtypename’ ];
$number=$_POST[ ‘borrownumber’ ];
$query_ins=mysql_query("insert into tb_readertype(name,number)
values( ‘$name’ ,’ $number’ )");
if($query_ins){
echo "<script language='javascript'>alert('添加成功! ');window.location.
href='/LibrarySys/readertype.php';</script>";
}else{
echo "<script language='javascript'>alert('添加失败! ');window.location.
href='/LibrarySys/readertype.php';</script>";
}
?>
```

添加读者类型的界面效果如图18.16所示。

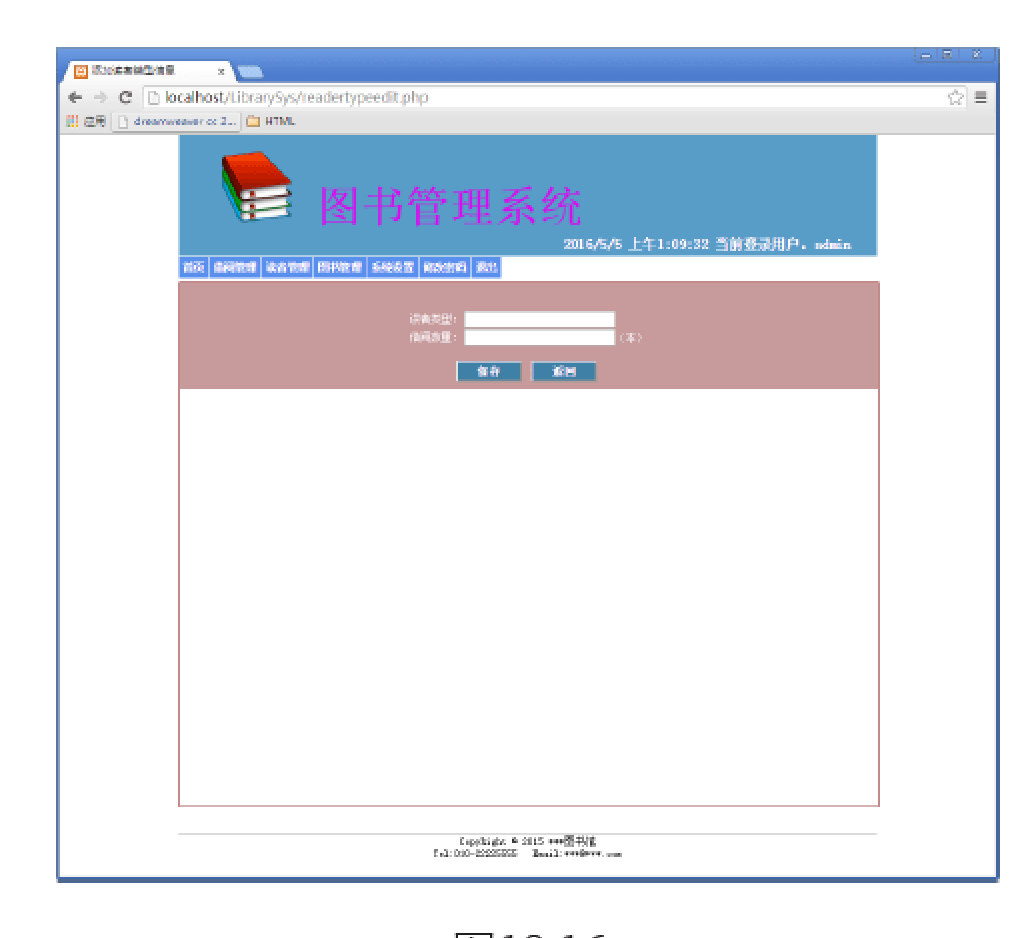


图18.16

```
    readertypeedit.php
    <?php session_start();?>
    <!doctype html>
    <html>
    <head>
    <meta charset="gb2312">
    <title>添加读者类型信息</title>
    <link type="text/css" href="/LibrarySys/css/style.css" rel="stylesheet"/>
    <script language="javascript">
    function check(form){
    if (form.readtypename.value==""){
    document.getElementById("errorMessage").innerHTML="请输入类型名称!";
    form.readtypename.focus();
    return false;
    }
    if (form.borrownumber.value==""){
    document.getElementById("errorMessage").innerHTML="请输入借阅数量!";
    form.borrownumber.focus();
    return false;
    }
    }
    </script>
    </head>
    <body>
    <?php
    include("navigation.php");
    include("conn.php");
    $action="";
    if($_GET[ ‘id’ ]==""){
    $action="readertypeadd.php";
     }else{
     $sql=mysql_query("SELECT * FROM tb_readertype where id=’ ".$_
GET[ ‘id’ ]."’ ");
     $info=mysql_fetch_array($sql);
     $action ="readertypeupdate.php";
     }
     ?>
    <div id="edit">
       <form id="readertypeform" name="readertypeform" action="<?php echo
$action ?>" method="post">
        <table>
       <tr><td>读者类型：</td><td><input type="text" id="readtypename"
name="readtypename" value="<?php if($_GET['id']!="") {echo $info['name'];}
else {echo "";} ?>"/></td></tr>
               <tr><td>借阅数量：</td><td><input type="text" id="borrownumber"
name="borrownumber"  value="<?php if($_GET['id']!="") {echo $info['number'];}
else {echo "";} ?>"/>（本）</td></tr>
            <tr><td colspan="2" align="center"><input class="btn" type="submit"
onClick="return check(readertypeform)" value="保存" /> <input class="btn"
type="button" value="返回" onClick="window.location.href='/LibrarySys/
readertype.php'" /></td></tr>
```

```
        <tr><td colspan="2" align="center"><label id="errorMessage"
name="errorMessage" ></label><input type="hidden" name="id" value="<?php if($_
GET[ ‘id’ ]!="") {echo $info[ ‘id’ ];} else {echo "";} ?>"/></td></tr>
        </table>
      </form>
    </div>
    <?php
    include("footer.php")
    ?>
    </body>
    </html>
```

编辑读者类型的界面效果如图18.17所示。

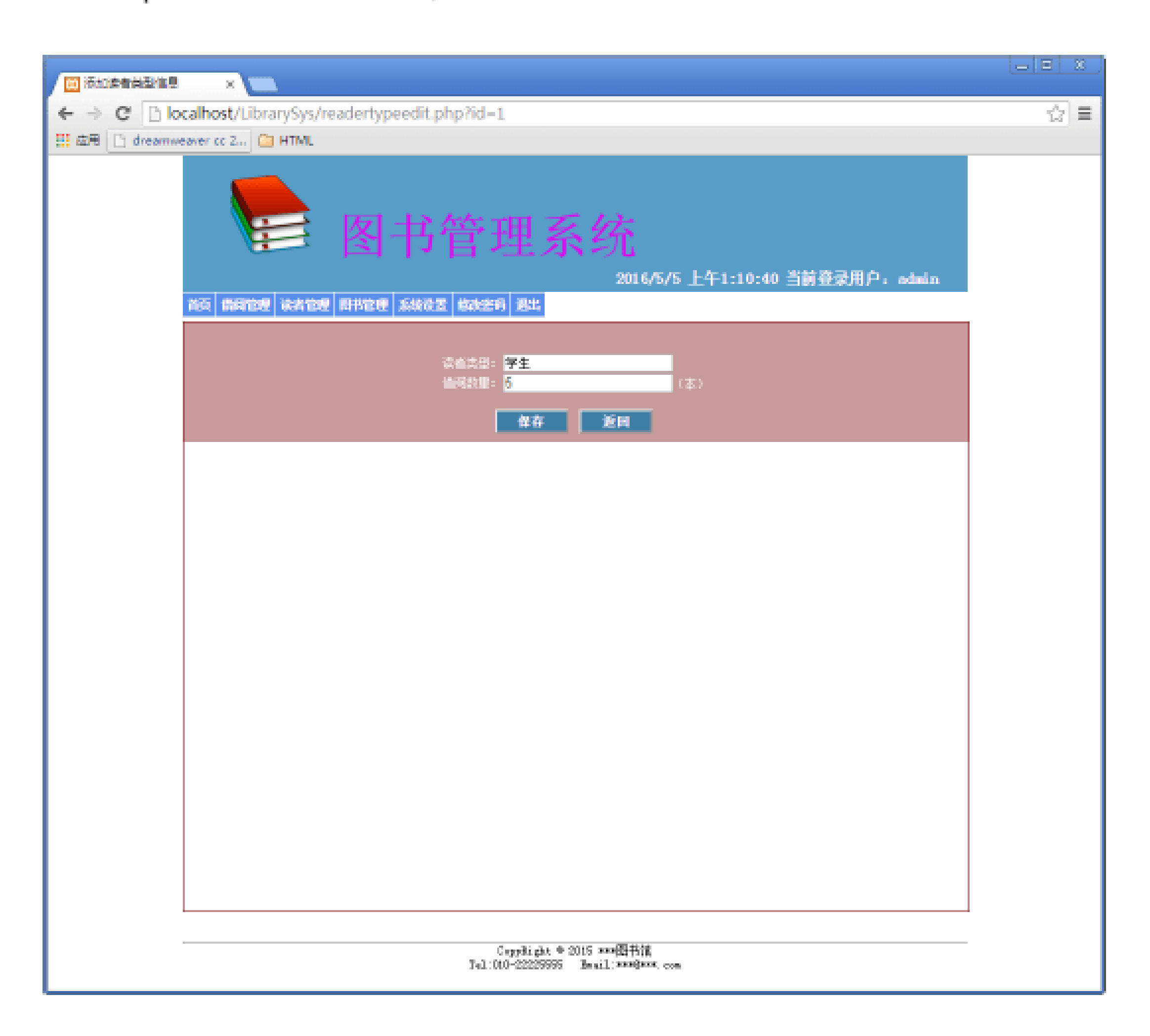


图18.17

```
    readertypedel.php
    <meta charset="gb2312">
    <?php
    include("conn.php");
    $id=$_GET['id'];
    $result=mysql_query("delete from tb_readertype where id='".$id."'");
    if($result==true){
    echo "<script language='javascript'>alert('删除成功! ');window.location.
href='/LibrarySys/readertype.php';</script>";
    }else{
    echo "<script language='javascript'>alert('删除失败! ');window.location.
href='/LibrarySys/readertype.php';</script>";
    }
    ?>
    readertypeupdate.php
    <meta charset="gb2312">
    <?php
    include("conn.php");
    $id=$_POST['id'];
    $name=$_POST['readtypename'];
    $number=$_POST['borrownumber'];
    $result=mysql_query("update tb_readertype set name='".$name."',number='".$
```

```
number."' where id='".$id."'");
    if($result==true){
    echo "<script language='javascript'>alert('修改成功！');window.location.
href='/LibrarySys/readertype.php';</script>";
    }else{
    echo "<script language='javascript'>alert('修改失败！');window.location.
href='/LibrarySys/readertype.php';</script>";
    }
    ?>
```

2. 读者档案管理

读者档案管理与读者类型管理都属于基础信息管理，它们在页面展示和业务处理方面都是类似的，读者可以在本章资源中获取到相关代码，这里就不再赘述。该界面效果如图18.18所示。

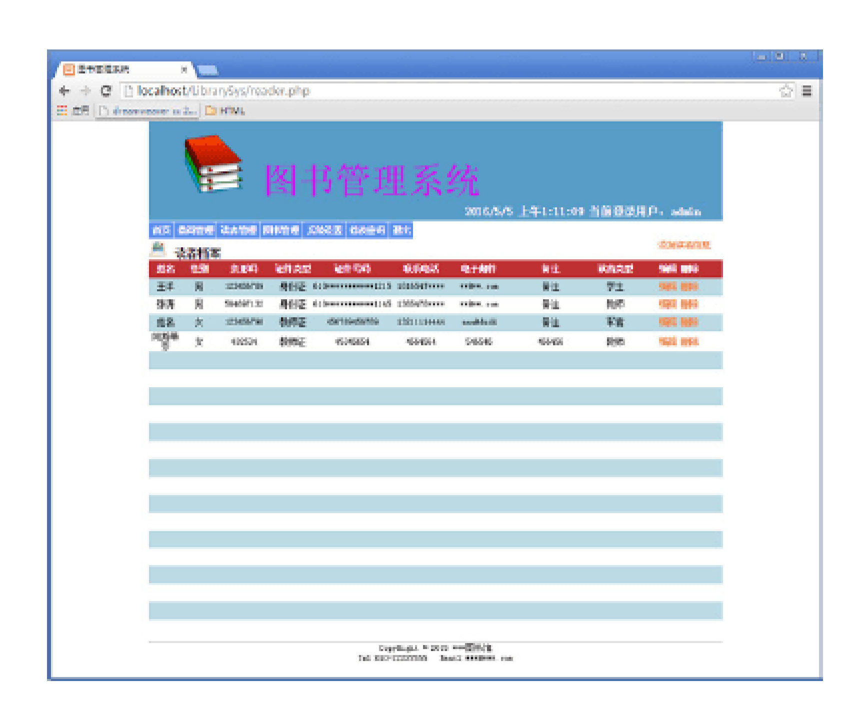


图18.18

18.2.6 其他基础信息管理

该系统中还有很多其他基础信息，它们的处理方式与读者管理界面类似，很多代码都可以拿来重复使用，这里就不再赘述，只展示相关界面的效果。如图18.19所示为图书管理界面，如图18.20所示为图书类型管理界面，如图18.21所示为出版社管理界面，如图18.22所示为用户管理界面。

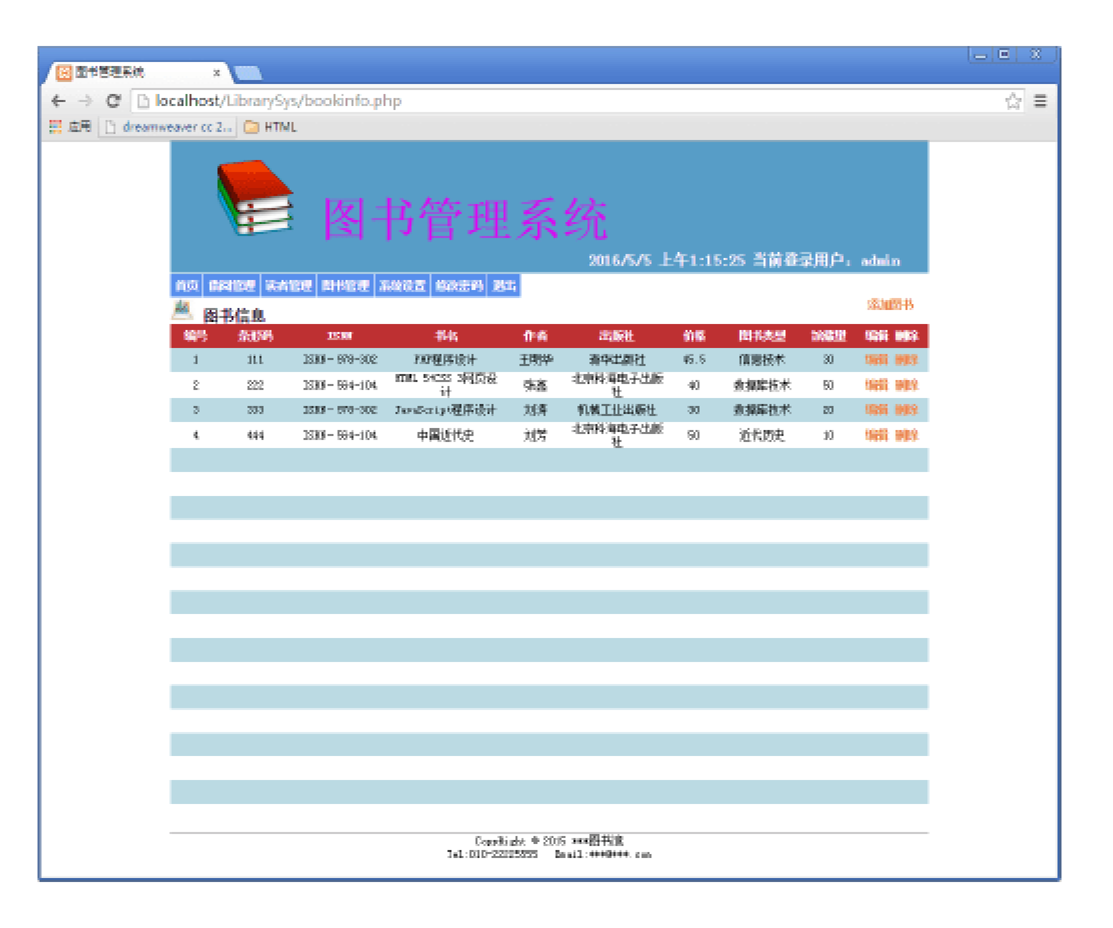


图18.19

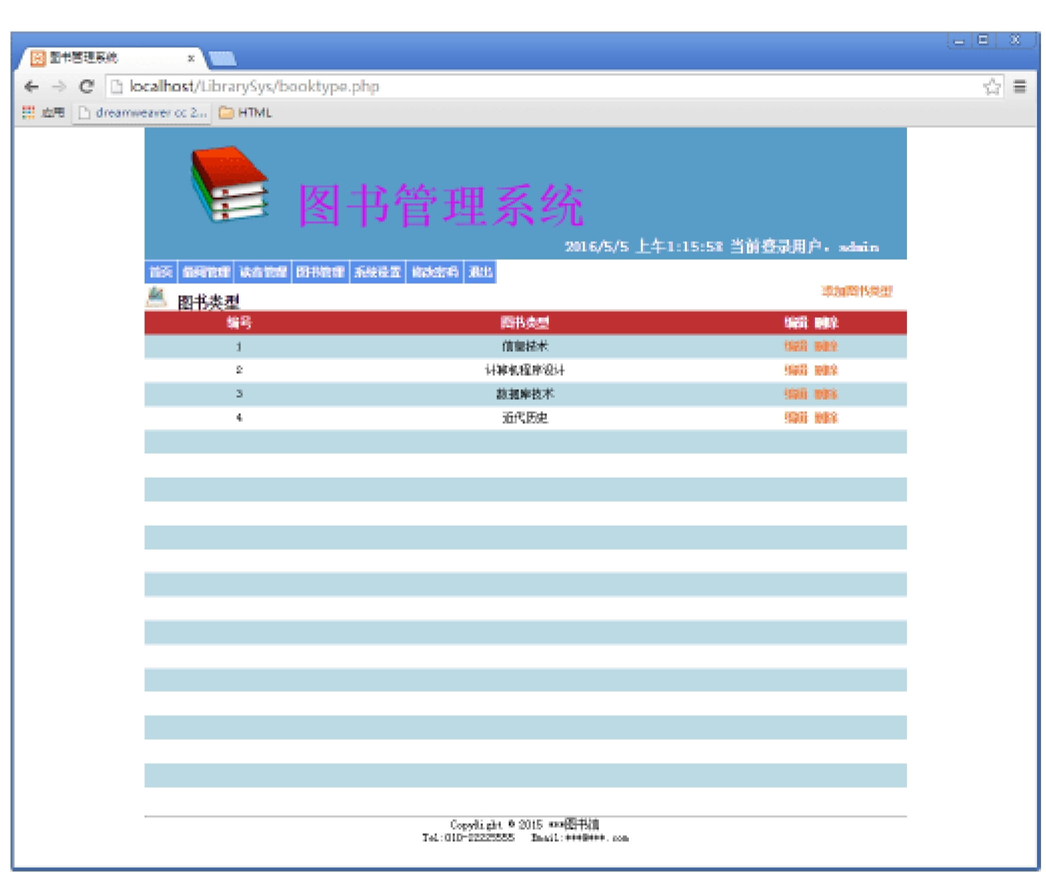


图18.20



图18.21



图18.22

18.2.7 修改密码

修改密码功能用于修改系统用户密码，页面passwordedit.php相关代码如下所示：

```
<?php session_start();?>
<!doctype html>
<html>
<head>
<meta charset="gb2312">
<title>添加读者类型信息</title>
<link type="text/css" href="/LibrarySys/css/style.css" rel="stylesheet"/>
<script language="javascript">
function check(form){
if (form.oldpwd.value==""){
document.getElementById("errorMessage").innerHTML="请输入旧密码!";
form.oldpwd.focus();
return false;
}
if (form.newpwd.value==""){
document.getElementById("errorMessage").innerHTML="请输入新密码!";
form.newpwd.focus();
return false;
}
if (form.confirmnewpwd.value==""){
document.getElementById("errorMessage").innerHTML="请输入确认密码!";
form.confirmnewpwd.focus();
return false;
}
if (form.newpwd.value!= form.confirmnewpwd.value){
document.getElementById("errorMessage").innerHTML="两次输入的新密码不一致!";
form.newpwd.focus();
return false;
```

```
}
}
</script>
</head>
<body>
<?php
include("navigation.php");
include("conn.php");
 ?>
<div id="edit">
   <form id="passwordupdateform" name="passwordupdateform" action="/LibrarySys/passwordupdate.php" method="post">
    <table>
          <tr><td>用户名：</td><td><?php echo $_SESSION['admin_name'] ?></td></tr>
<tr><td>旧密码：</td><td><input type="password" id="oldpwd" name="oldpwd" /></td></tr>
          <tr><td>新密码：</td><td><input type="password" id="newpwd" name="newpwd"  /></td></tr>
          <tr><td>重复新密码：</td><td><input type="password" id="confirmnewpwd" name="confirmnewpwd" /></td></tr>
          <tr><td colspan="2" align="center"><input class="btn" type="submit" onClick="return check(passwordupdateform)" value="保存" /> <input class="btn" type="button" value="返回" onClick="window.location.href='/LibrarySys/index.php'" /></td></tr>
          <tr><td colspan="2" align="center"><label id="errorMessage" name="errorMessage" ></label><input type="hidden" name="id" value="<?php echo $_SESSION[‘admin_id’] ?>" /></td></tr>
    </table>
  </form>
</div>
<?php
include("footer.php")
?>
</body>
</html>
```

修改密码的逻辑处理在passwordupdate.php页面中完成，相关代码如下所示：

```
<meta charset="gb2312">
<?php
include("conn.php");
$id=$_POST['id'];
$oldpwd=$_POST['oldpwd'];
$newpwd=$_POST['newpwd'];
$sql=mysql_query("select * from tb_sysuser where id='".$id."'",$conn);
$info=mysql_fetch_array($sql);
if($info['pwd']!=$oldpwd){
echo "<script language='javascript'>alert('您输入的旧密码有误，请重新输入！');history.back();</script>";
return false;
```

```
    }
    $sql="update tb_sysuser set pwd='".$newpwd."' where id='".$id."'";
    $result=mysql_query($sql);
    if($result==true){
    echo "<script language='javascript'>alert('修改成功！');window.location.
href='/LibrarySys/index.php';</script>";
    }else{
    echo "<script language='javascript'>alert('修改失败！');window.location.
href='/LibrarySys/index.php';</script>";
    }
    ?>
```

修改密码的主界面效果如图18.23所示。

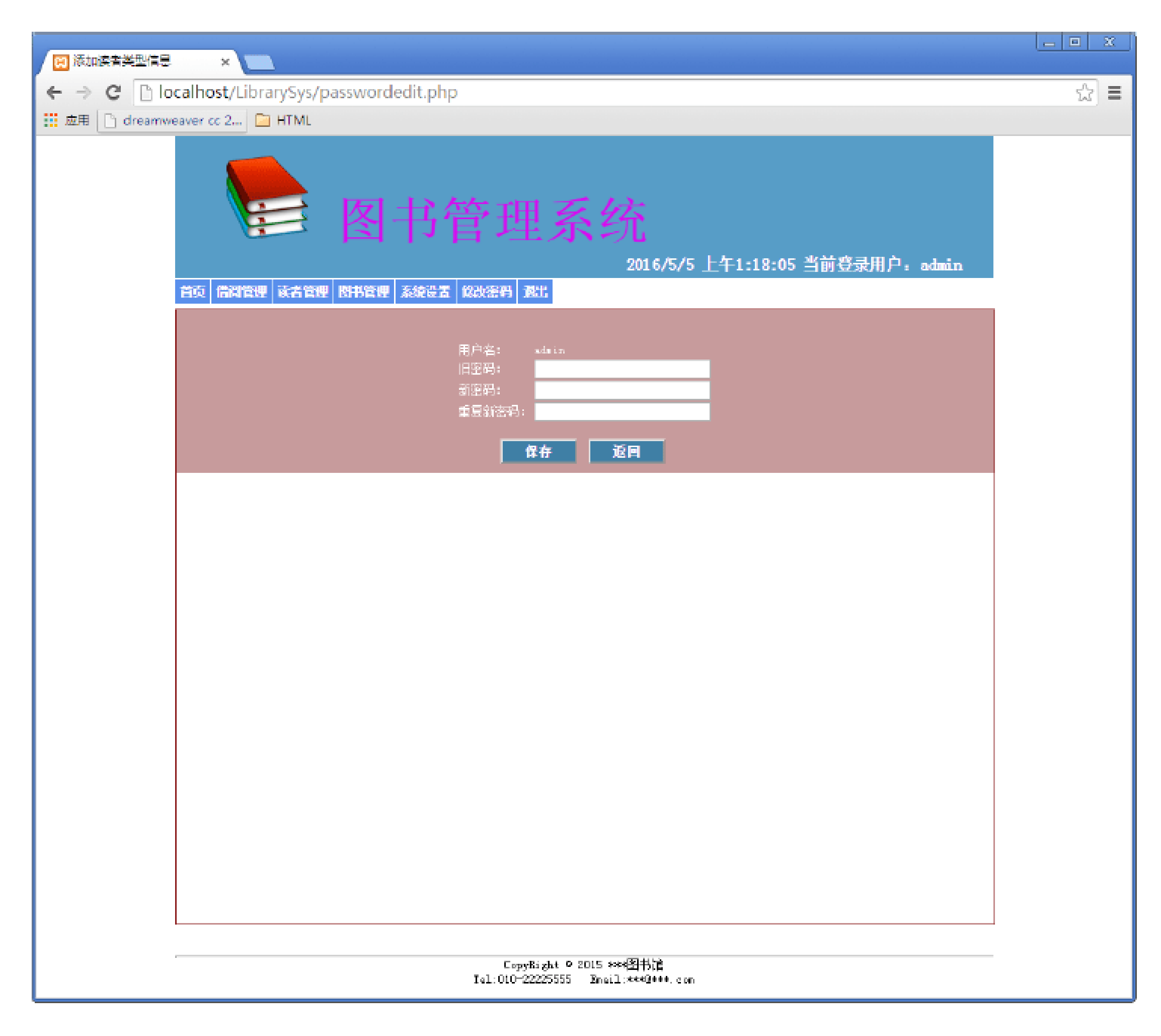


图18.23

18.2.8 退出功能

系统退出功能主要在logout.php页面中完成，该页面主要功能用于清除session和重定向到login页面，相关代码如下所示：

```
<?php
session_start();
session_unset();
session_destroy();
header("location:login.php");
?>
```

第19章 申请域名和空间

网站制作完成后，需要将网站上传到网络空间，并给网络空间绑定域名，这样其他人就可以在互联网上通过域名来访问这个网站了。



19.1 申请域名

一个好的域名不但方便记忆，而且对于网站后期的推广也有很好的作用。例如国内大家都熟知的百度（baidu.com）、腾讯（qq.com）、淘宝（taobao.com）、京东（jd.com）等。本节将详细介绍与域名有关的知识。

19.1.1 什么是域名

我们都知道，IP地址是互联网上区别每台电脑的唯一标识，但是IP地址比较长，不方便记忆，也不好区分，所以人们就给ip地址带了一个面具，这个面具就是域名。每个域名和一个IP地址绑定，当用户访问域名的时候，域名解析服务（DNS）就会将用户访问的域名自动解析为具体的IP地址。

域名由若干字母、数据或文字组成，在互联网上域名具有唯一性，如果要使用域名，必须先在域名服务商那里申请域名，域名的申请遵循先注册先得的原则。目前国际顶级域名，如.com、.net等简短好记的域名已经基本上都被注册了；而国内顶级域名，如.cn、com.cn等简短好记的域名也所剩不多了。

19.1.2 实例：申请域名

下面我们以在万网注册域名为例，详细介绍申请域名的整个过程，具体步骤如下所示：

01 在浏览器中打开万网地址http://wanwang.aliyun.com/，如图19.1所示，在网页的左下角点击“域名服务”。



图19.1

02 在新打开的网页中，我们可以看到一个很大的输入框，在这个输入框中输入要注册的域名，例如输入xinwangmeng，如图19.2所示。然后点击后面的“查域名”按钮，就可以查询到当前这个域名的相关信息。



图19.2

03 在图19.3中显示了查询结果，我们可以看到xinwangmeng.com这个国际顶级域名已经被其他人注册了，而且正在二次销售。所以，如果我们一定要用这个域名，就必须向域名目前的所有者购买，如果不是必须的，我们还可以选择后缀为.cn、.net等的其他域名。

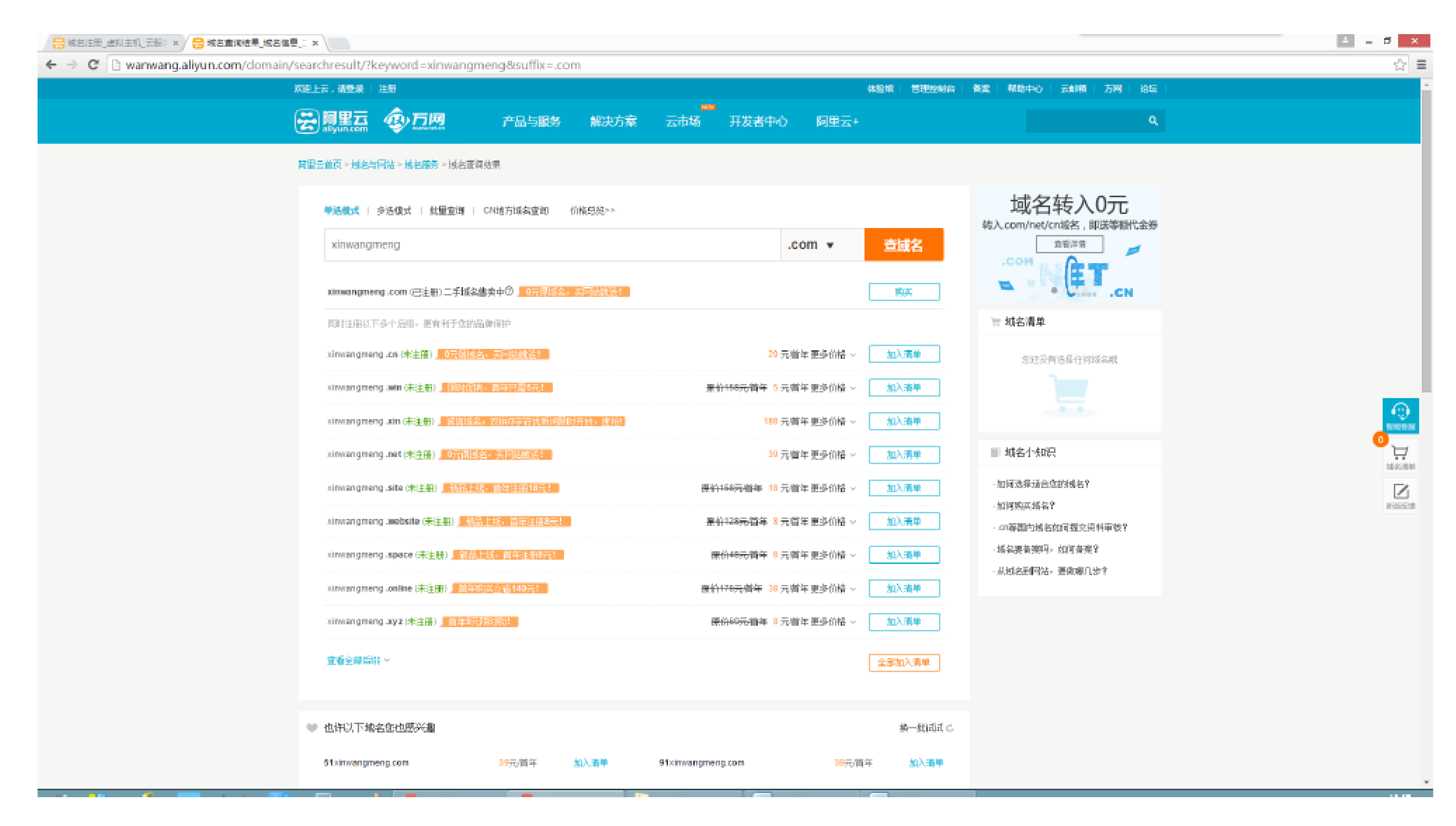


图19.3

04 我们以后缀为.cn为例，可以看到该域名对应的价格是29元/首年，点击“加入清单”按钮，可以在右边的侧边栏中看到域名清单列表中已经添加了一个域名，如图19.4所示。

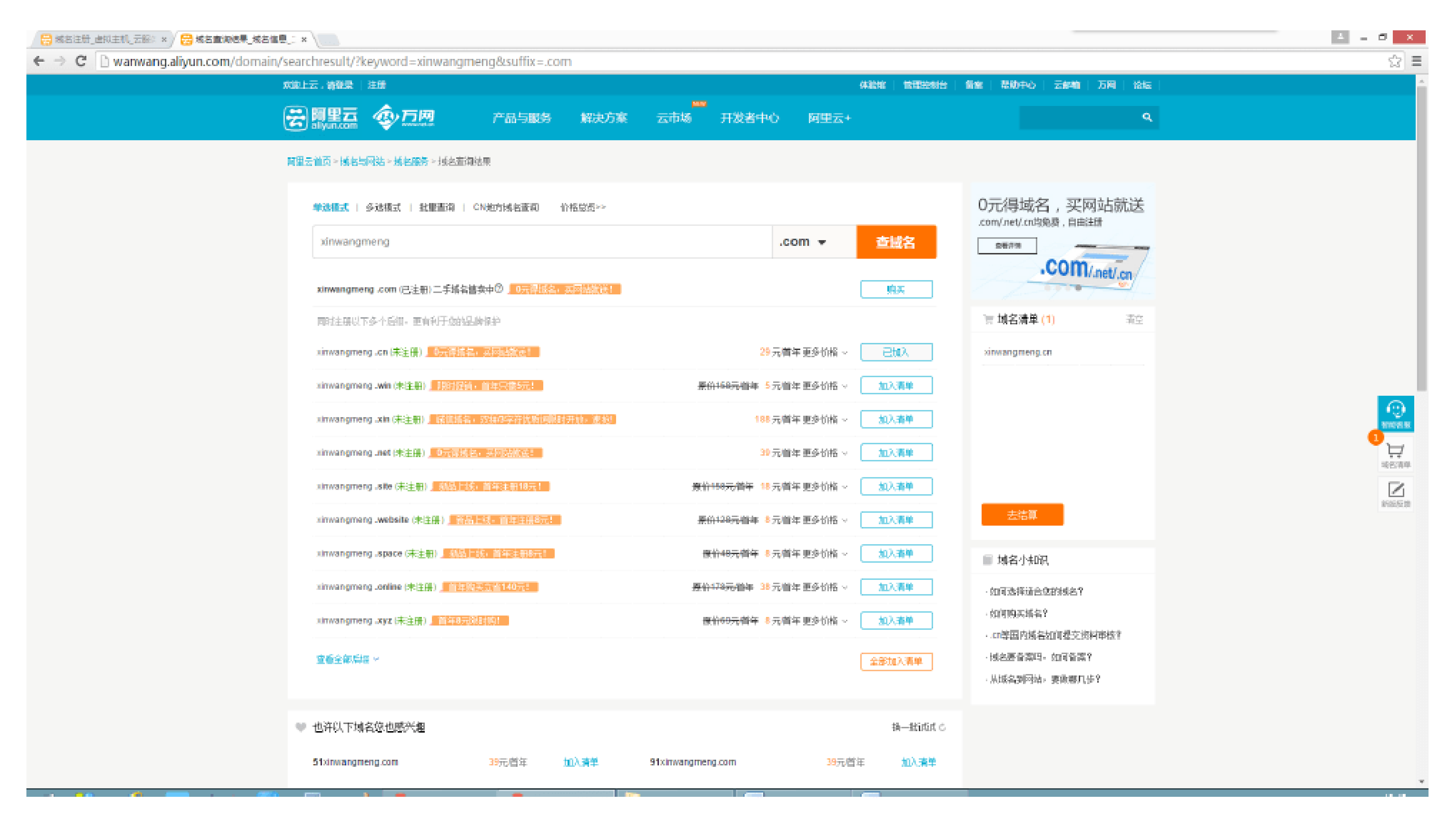


图19.4

05 点击“去结算”按钮，打开购物车列表页面，如图19.5所示，在该页面中选择域名的购买年限为1年。后期还需要用户上传相关资料进行备案，所以这里需要确定当前申请的域名将用于个人还是企业，这里我们选择“个人”，最后点击“立即购买”。



图19.5

06 注意在购买万网域名之前，需要注册阿里云账户，登录后系统会跳转到结算页面，要求用户填写该域名相关资料。如图19.6所示，按照页面要求填写相关资料。

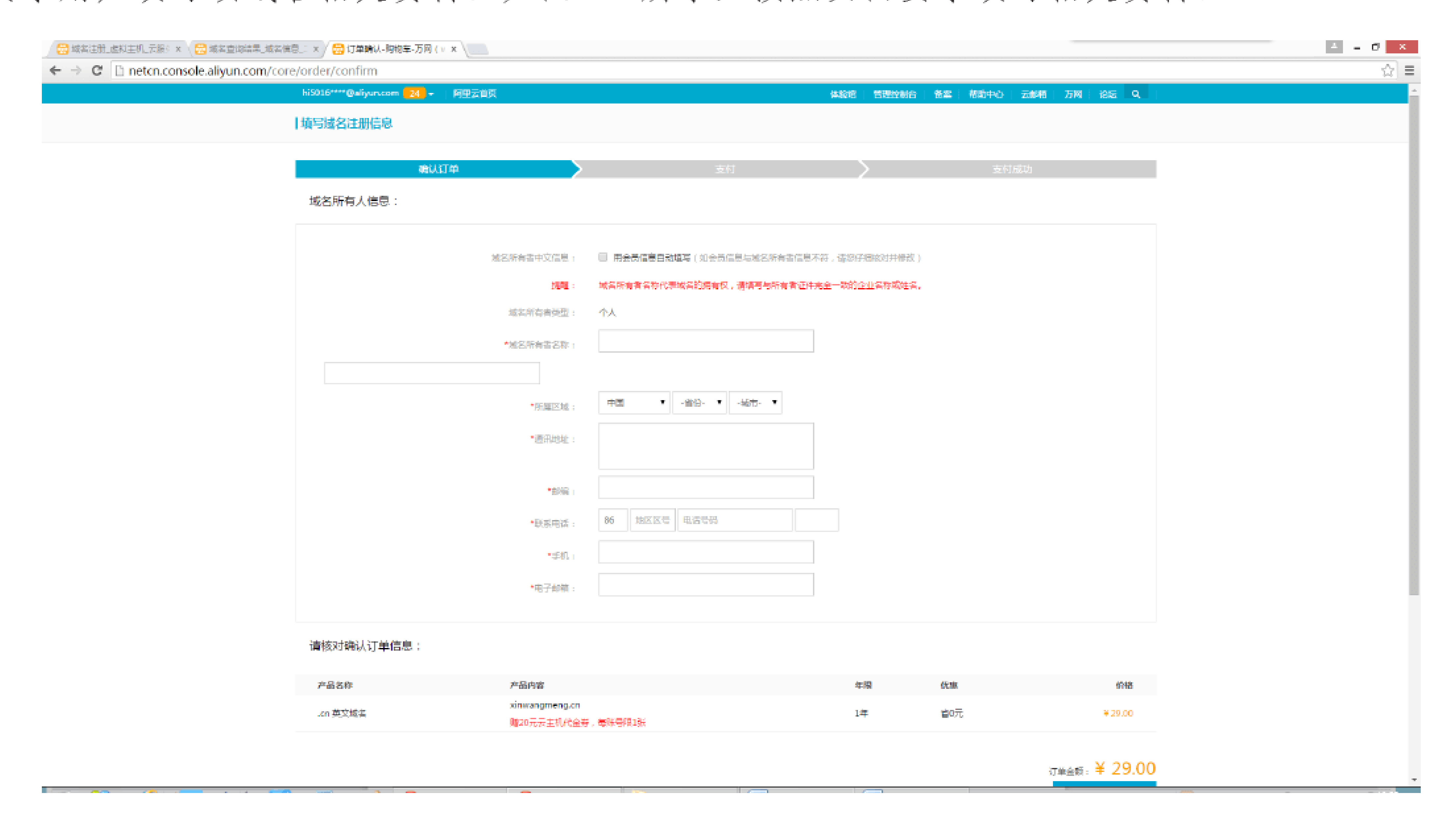


图19.6

07 最后点击“结算”按钮，进入支付界面，如图19.7所示，选择方便的支付方式进行支付，即可完成域名的申请。

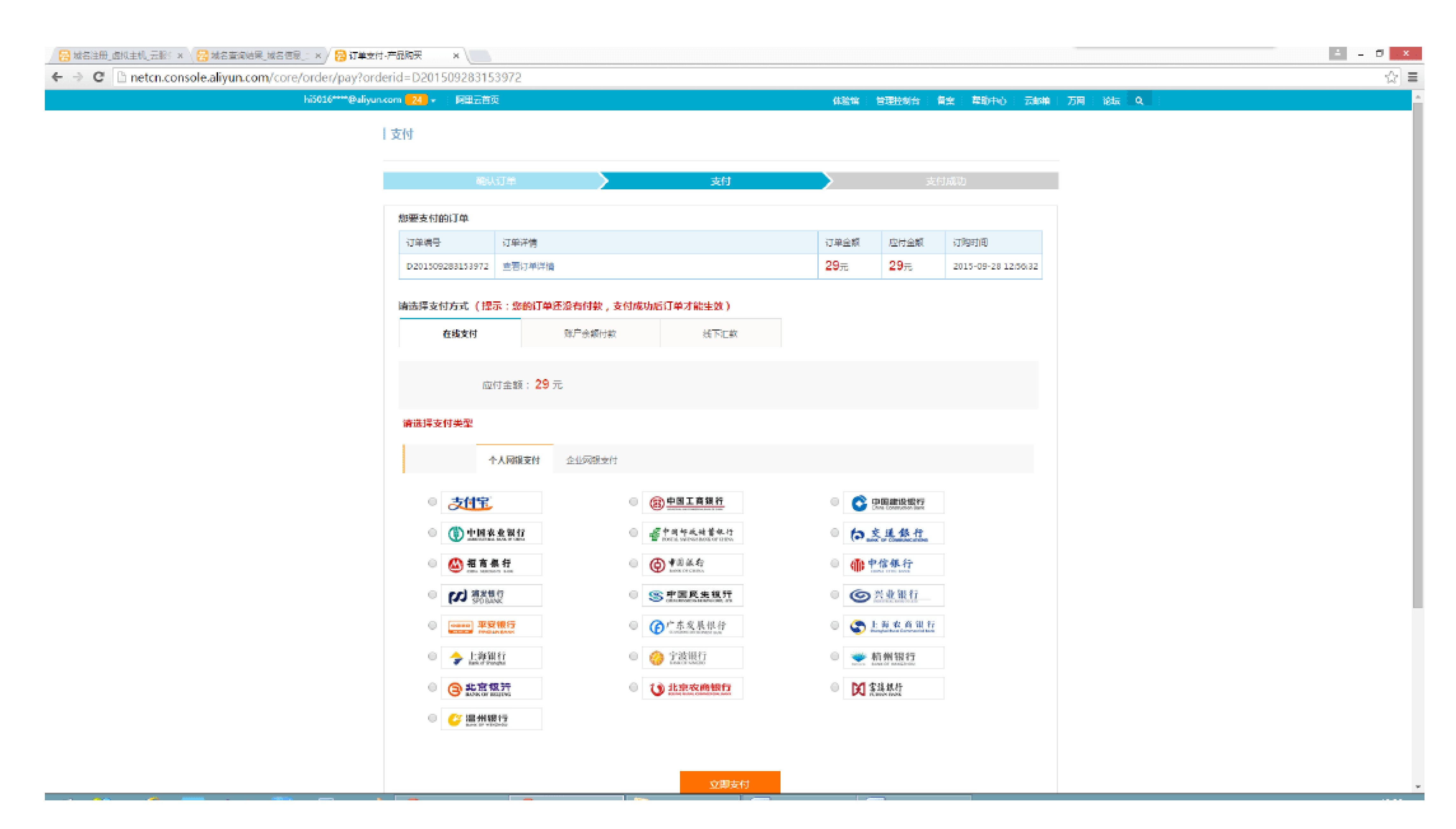


图19.7



19.2 申请网站空间

网站空间的质量直接决定了网站访问的速度和稳定性，所以在申请网站空间的时候，一定要选择一个速度快并且性能稳定的服务提供商。

19.2.1 网站空间简介

网站空间就是存放网站内容的空间。当然也可以将网站存放在自己的电脑上，然后向外发布程序，其他人可以通过自己电脑的ip地址访问到发布的网站。但是多数情况下，我们自己的电脑配置都不高，而且不能保证全天候开机，所以最好还是将网站存放在专业的网站空间中。

网站空间也叫做虚拟主机空间，是专业存储网站内容的地方，按照网站空间的形式，可以分为虚拟空间、合租空间和独立空间。虚拟空间是目前使用最多的一种形式，不仅成本低廉，而且提供技术支持和空间维护；而合租空间和独立主机则适用于中型或大型网站，他们需要更多的空间和更稳定的运行。

19.2.2 实例：申请网站空间

下面同样以申请万网的网站空间为例，详细介绍申请网站空间的过程。

（1）打开万网主页http://wanwang.aliyun.com/，如图19.8所示，可以看到主机服务一栏，点击“主机服务”。



图19.8

（2）打开虚拟主机申请页面，如图19.9所示。在这个页面中，我们可以看到多种类型的网站空间，每个网站空间的配置都不一样，所以购买的价格也不一样，配置越高的空间价格越高。

（3）以共享经济版的网站空间为例，点击“立即购买”按钮，打开主机购买界面，如图19.10所示。在该页面中选择产品名为“共享经济版”，机房默认为“青岛机房”，操作系统选择Windows，其他基本配置是服务提供商为这款产品设置的基本配置，购买时长选择“1年”，然后点击“立即购买”按钮。



图19.9

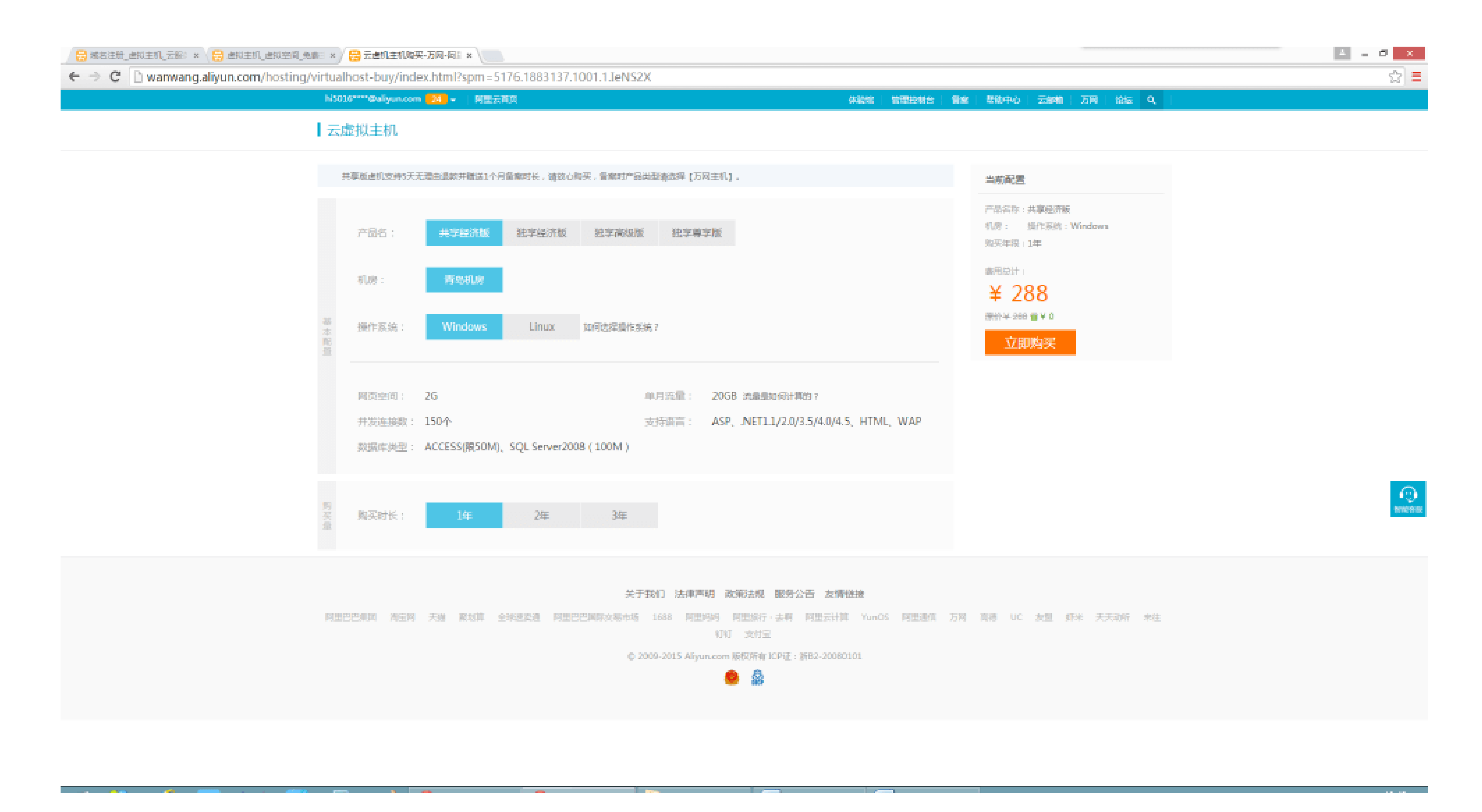


图19.10

（4）打开支付页面，如图19.11所示，完成剩余的支付流程，这样就完成了网站空间的申请操作。

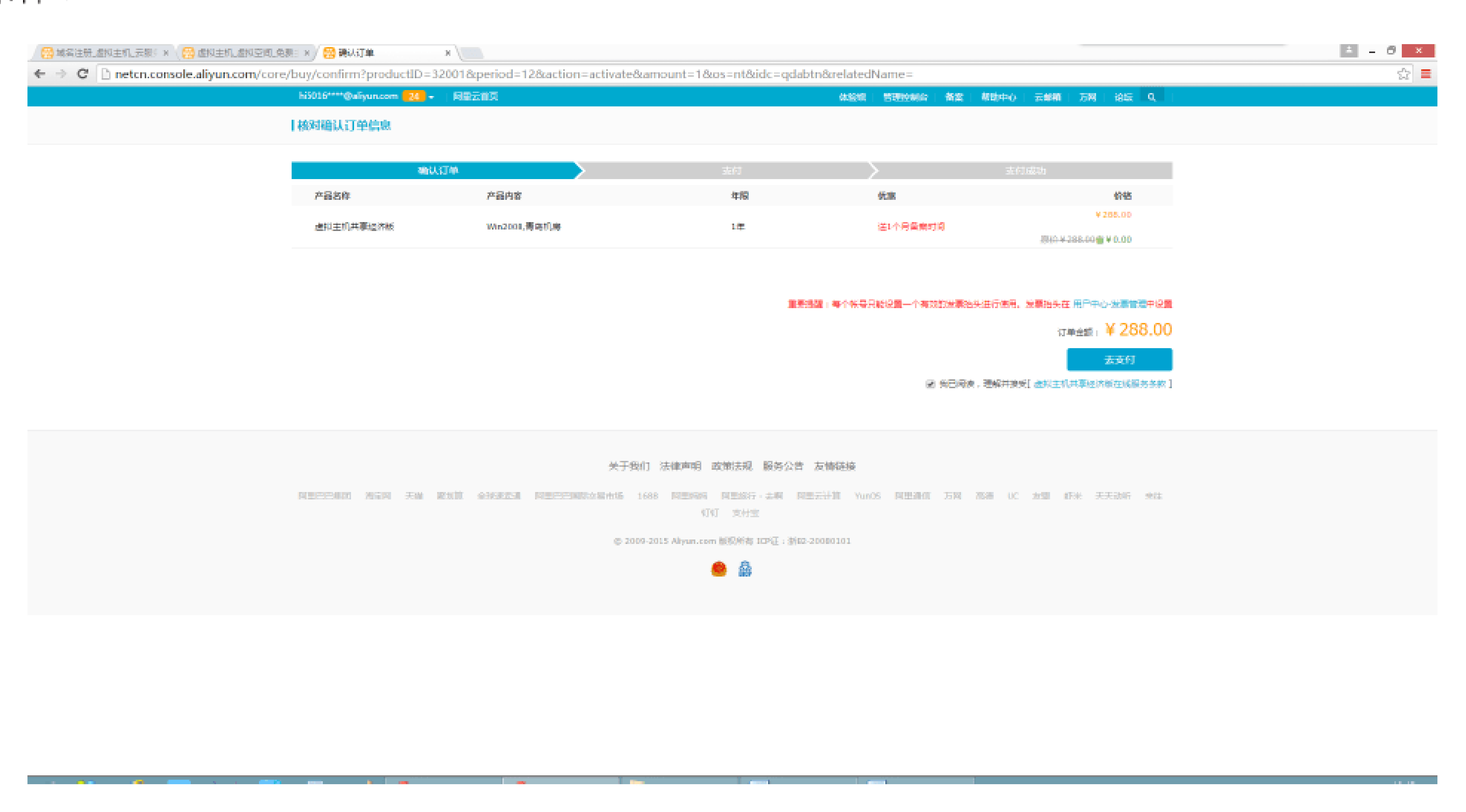


图19.11



19.3 绑定域名和空间

申请了域名和网站空间之后，还需要在空间提供商的管理平台上进行域名和空间的绑定操作，这样在互联网上访问域名的时候才能访问到网站空间的内容。点击万网首页上的“管

理控制台”链接，如图19.12所示。



图19.12

打开如图19.13所示的页面，在该页面左侧的导航栏中点击“云虚拟主机”菜单。

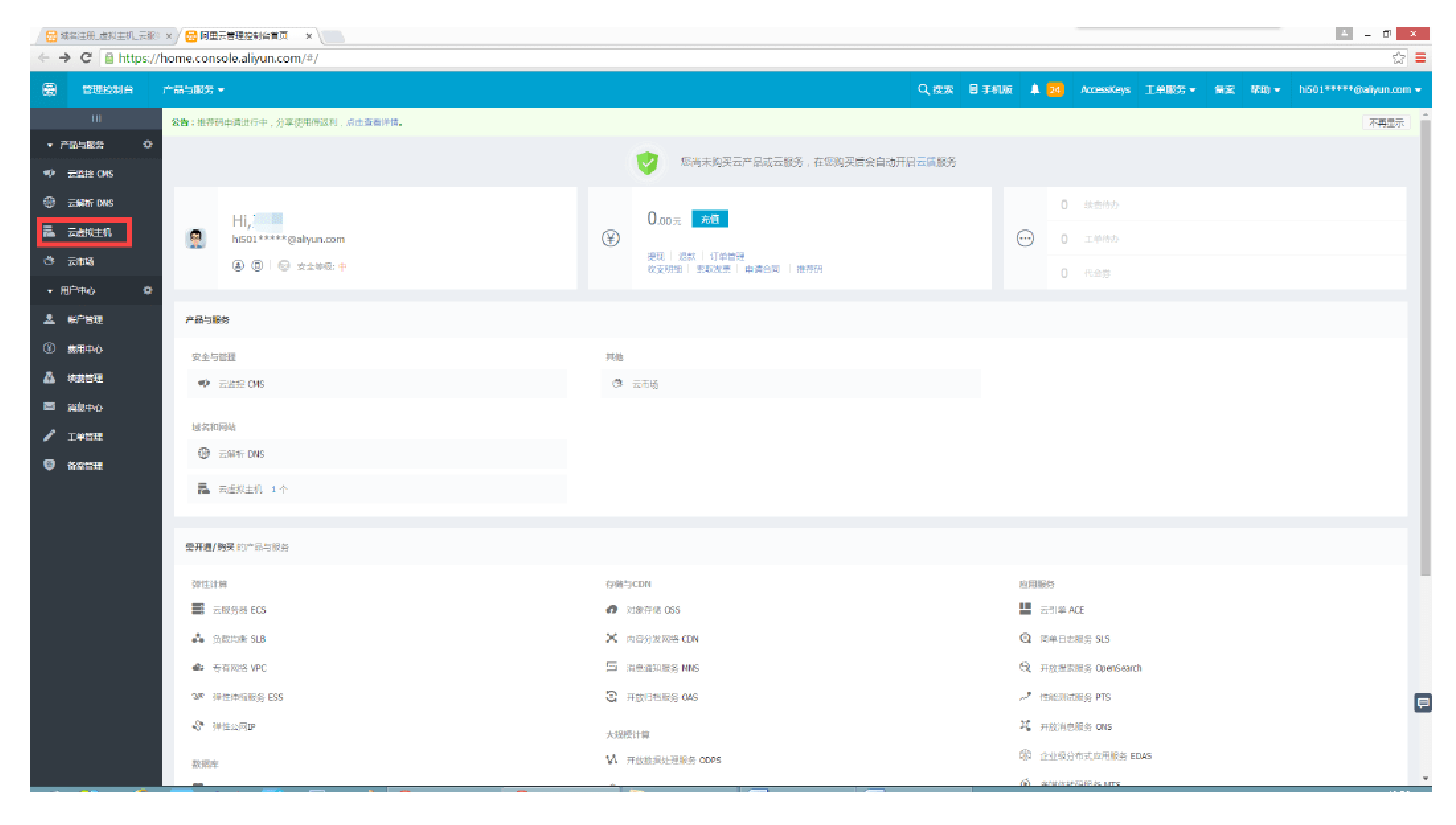


图19.13

打开如图19.14所示的页面，该页面中已经有一个虚拟主机列表，在最右侧的红色区域中点击“管理”，打开虚拟主机管理界面。

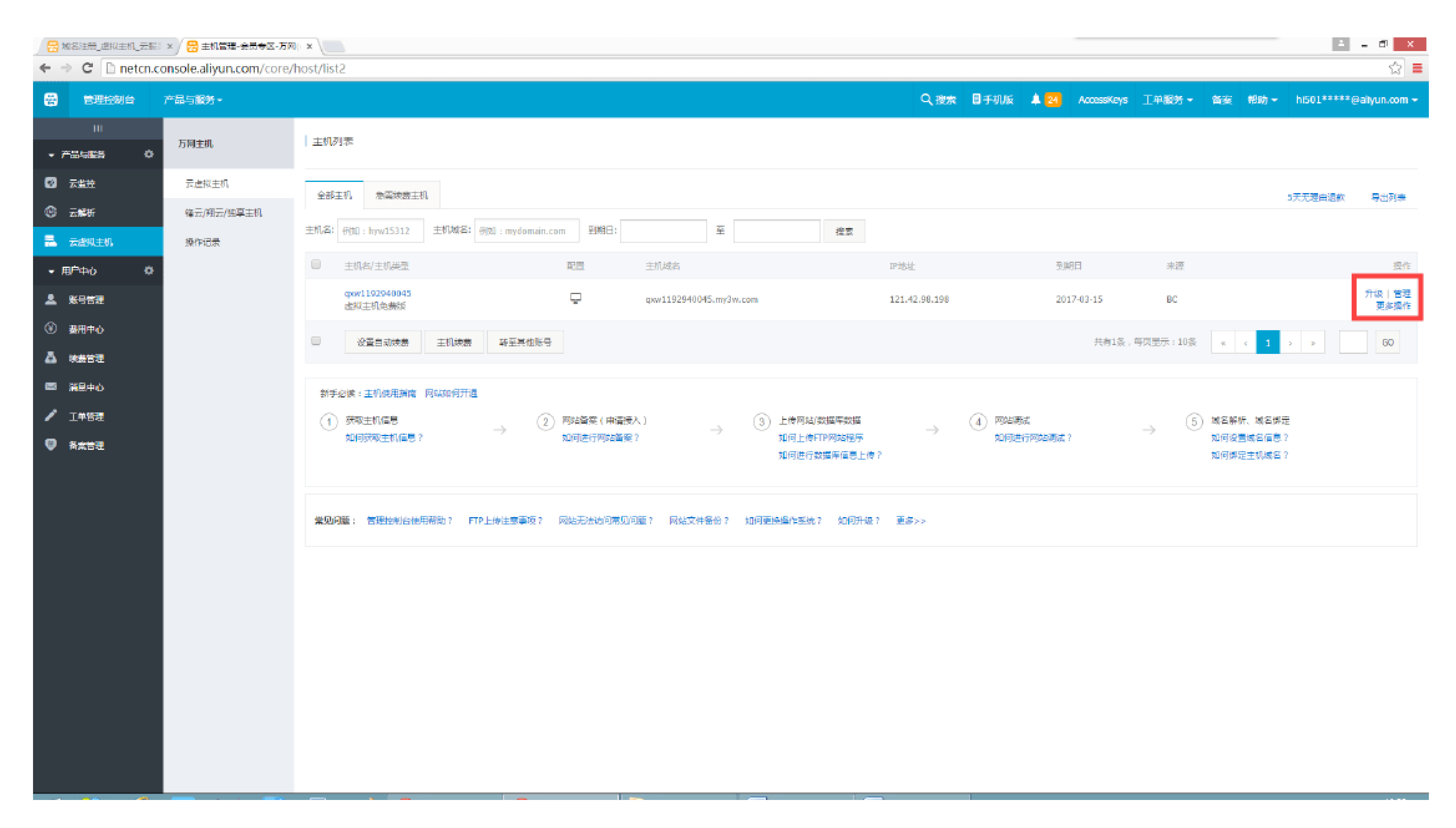


图19.14

如图19.15所示，在该页面的左侧列表中选择“基础环境设置”菜单，选择“域名绑定”子菜单，就可以看到右侧页面，选中“绑定域名”选项，输入新的域名，然后单击“添加”按钮，即可将输入的域名与当前主机绑定到一起。



图19.15

第20章 测试、上传与维护网站

网站开发完成后，需要对网站进行一系列的测试，保证网站能正常运行。首先进行本地测试然后将网站上传至网站空间，进行在线测试。另外，网站内容需要及时维护和更新，从而保证网站健康运行。



站点的测试

网站开发完成后，先在本地站点发布，通过本地测试是检验网站的第一个步骤。网站测试的方法和内容很多，下面介绍几种常见的测试方法。

20.1.1 功能测试

功能测试又称为黑盒测试或数据驱动测试，它把测试对象看作一个黑盒子。利用黑盒测试法进行动态测试时，只需要测试软件产品的功能，而不需要知道其内容结构和处理过程。

功能测试主要是为了检测软件产品是否存在功能性错误或遗漏，软件界面是否存在错误，数据结构或外部数据库访问是否错误，性能是否满足要求，以及初始化和终止功能是否正常等。

下面介绍一些常用的功能测试方法。

（1）相关性检查：检查在界面上进行增删改操作是否会影响其他内容的变化，这些变化是否正常。

（2）检查相关按钮的功能：每一个按钮都有自己对应的功能，检查这些按钮对应的功能是否正确。

（3）超链接检查：检查界面上的每个超链接是否有对应的页面，点击超链接后切换页面是否正确。

（4）字符串长度检查：检查界面输入框中能输入的字符串是否有长度限制，尝试输入最大长度限制的字符串，检查是否会引发错误。

（5）输入类型检查：检查界面输入框中的内容是否有类型要求，如年龄只能输入数字，如果输入其他类型将引发错误。

（6）信息唯一性检查：在界面输入的信息中，是否存在唯一性要求，如账号等，如果有检查其唯一性。

（7）按条件查询检查：在按条件查询的界面中，输入不同的查询条件，检查输出的内容是否正确。

（8）必填项检查：检查界面中多个提交信息中，所有的必填项是否都有数据，如果没有，在提交操作之前是否有提示。

（9）重复提交表单：检查在成功提交表单之后，点击后退按钮再次提交，系统对二次提交的表单如何处理。

（10）中文字符处理：在可以输入中文的系统输入中文，检查是否会出现乱码。

（11）上传下载文件检查：检查上传与下载文件的功能是否能够实现，上传文件是否能够浏览。对上传文件的格式是否有规定，系统是否有解释信息。

20.1.2 浏览器兼容性测试

浏览器是Web系统中一个非常重要的组成部分，它将直接与用户进行交互。同一个网站在不同的浏览器上可能会有不同的效果，而目前用户可以选择的浏览器却非常多，所以，为了保证大多数用户都能正常浏览网站，我们有必要对浏览器的兼容性进行测试。

虽然目前市面上的浏览器种类非常多，但是影响浏览器兼容性的主要原因还是在于浏览器的内核。所谓浏览器的内核，就是指它的渲染引擎。目前市面上比较流行的浏览器内核可以分为以下几种。

（1）Trident：大名鼎鼎的IE浏览器使用的就是Trident内核，它也是目前市面上最流行、用户数量最多、使用范围最广的一个浏览器内核。国内很多的浏览器也是基于Trident内核开发的，如遨游浏览器、搜狗浏览器、世界之窗浏览器等等。

（2）Gecko：Gecko是一款用C++编写的渲染引擎，它的源码是开放的。火狐浏览器使用的就是Gecko内核。

（3）Webkit：Webkit是Mac OS X v10.3及以上版本所包含的软件框架，它是Mac OS X的Safari网页浏览器的基础。谷歌浏览器使用的就是Webkit内核。

（4）Presto：Presto是一个由Opera Software开发的浏览器排版引擎，它取代了旧版Opera中所使用的Elektra排版引擎，加入了动态功能。例如，网页或其他部分可随着DOM及Script语法的事件而重新排版。

在对网站进行浏览器兼容性测试的时候，只需要根据浏览器的内核选择几种浏览器进行测试即可。另外还需要考虑到当前比较流行的几种浏览器，例如，IE仍然是使用最多、使用范围最广的浏览器，而火狐浏览器和谷歌浏览器也是目前使用比较广泛的浏览器。

20.1.3 超链接的测试

每一个网站都是由很多的网页组成的，这些网页之间通过超链接相互关联，组成一个庞大的关系网。超链接测试的内容就是检查这些网页之间的超链接是否有效，链接对象是否正确，是否有某些页面被遗漏等。

如果网站页面比较少，可以通过人工的方式进行测试，但是对于大型网站来说，组成网站的页面可能有成百上千个，这时对超链接的测试将是一个非常庞大的工作。

大量测试工作的出现，衍生出了一些测试工具，如Html link validator和Xenu Link Sleuth等，使用这些工具可以对本地和网站上的网页文件链接进行检测，它们可以检测出网站上活动的链接和死链接。

20.2 站点的上传

站点上传是指将站点下的网站上传至网络空间的过程，可以使用Dreamweaver或上传工具完成站点上传工作。

20.2.1 使用Dreamweaver上传

Dreamweaver是制作网站的一款利器，它不仅可以编写网站，还可以将网站上传至网络空间，具体操作步骤如下：

01 打开Dreamweaver软件，选择站点，新建站点命令，打开如图20.1所示界面，在该界面中输入站点名称和保存文件的位置。

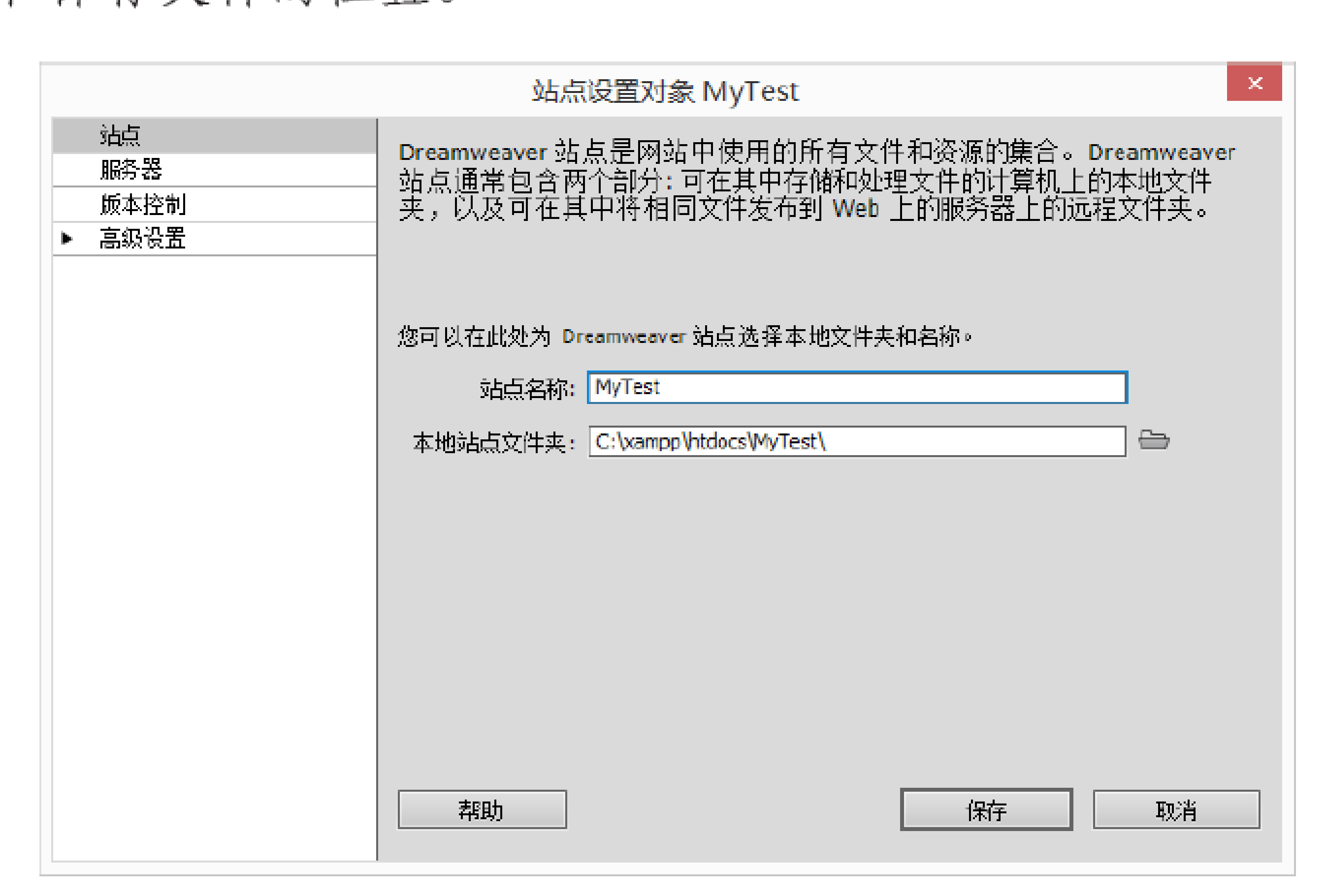


图20.1

02 在上图左侧的列表中选择“服务器”选项，如图20.2所示。

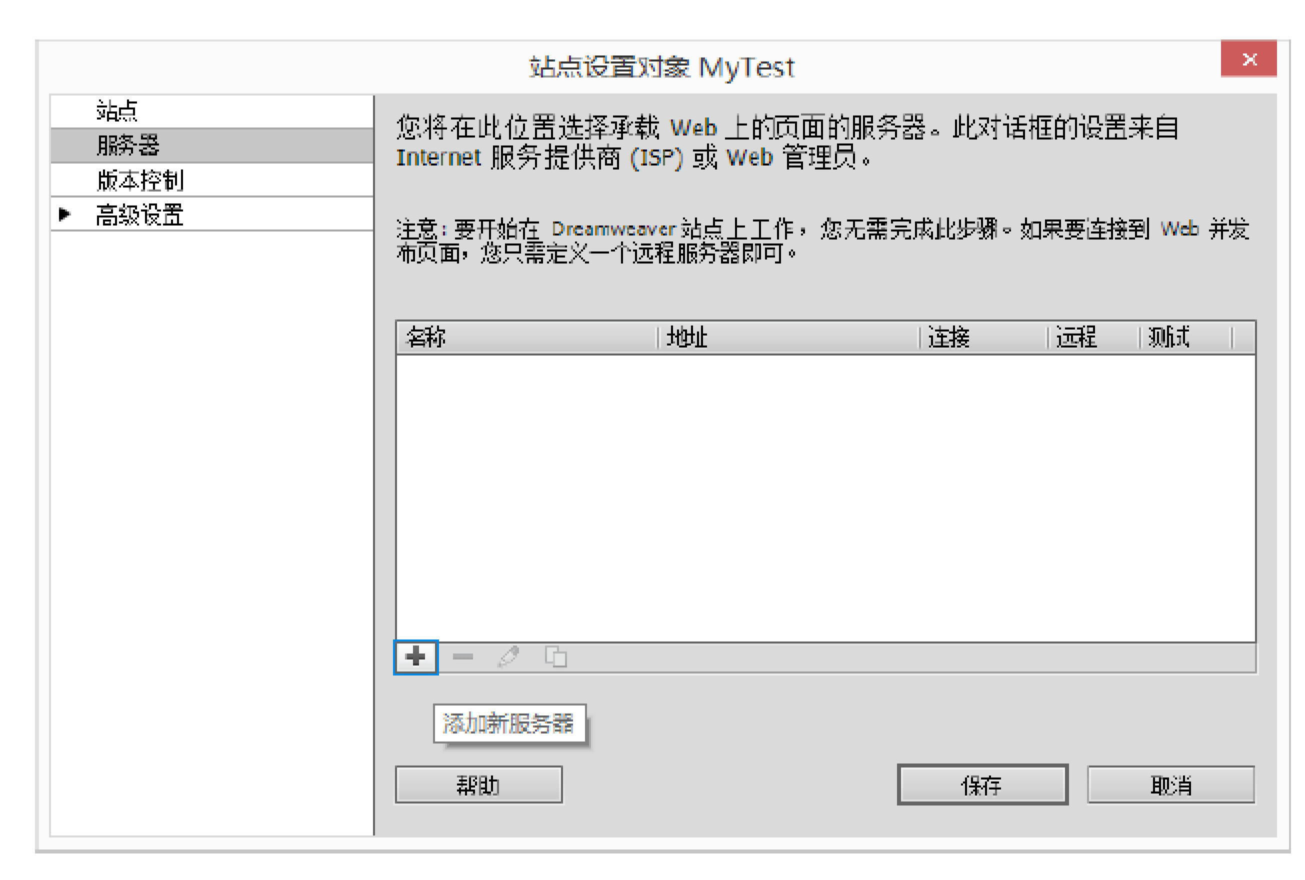


图20.2

03 点击添加新服务器的加号，打开服务器设置对话框，如图20.3所示。在该对话框中设置服务器的名称，选择连接方法为FTP。你可以从网络空间服务器的管理界面中获取FTP地址、用户名和密码信息，如图20.4所示。

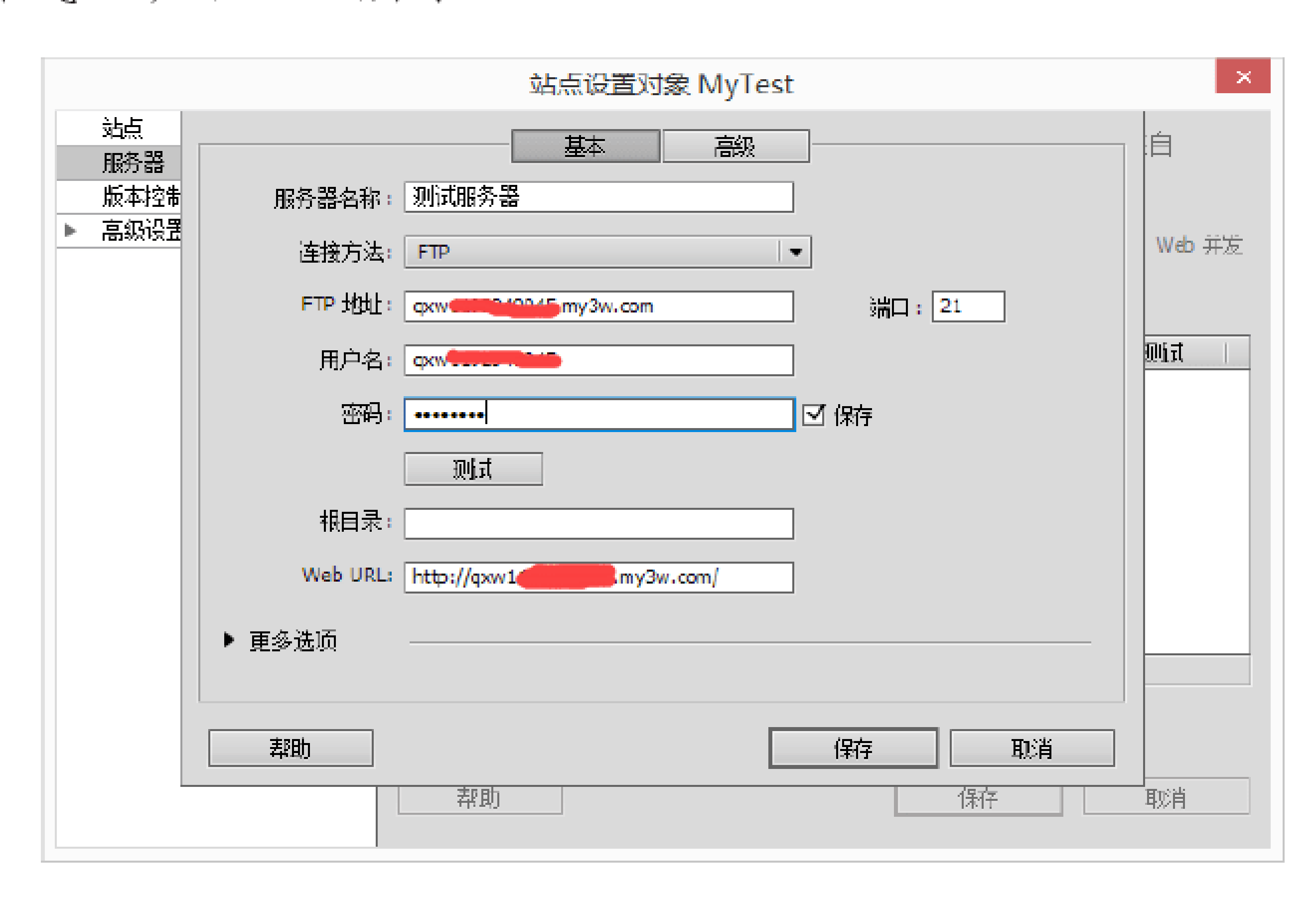


图20.3

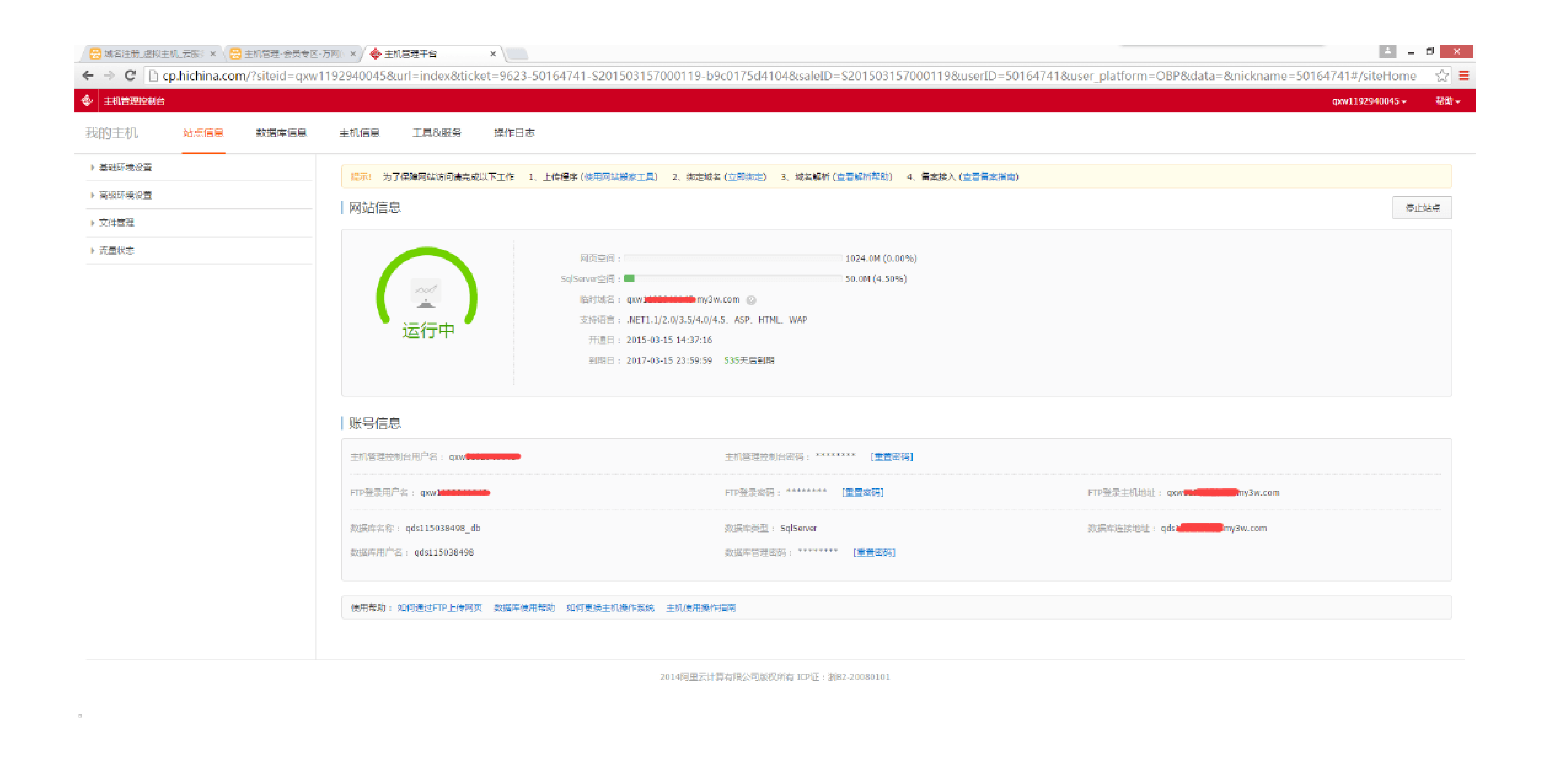


图20.4

04 填写完所有信息后，单击“测试”按钮，如果测试成功，则弹出如图20.5所示效果。

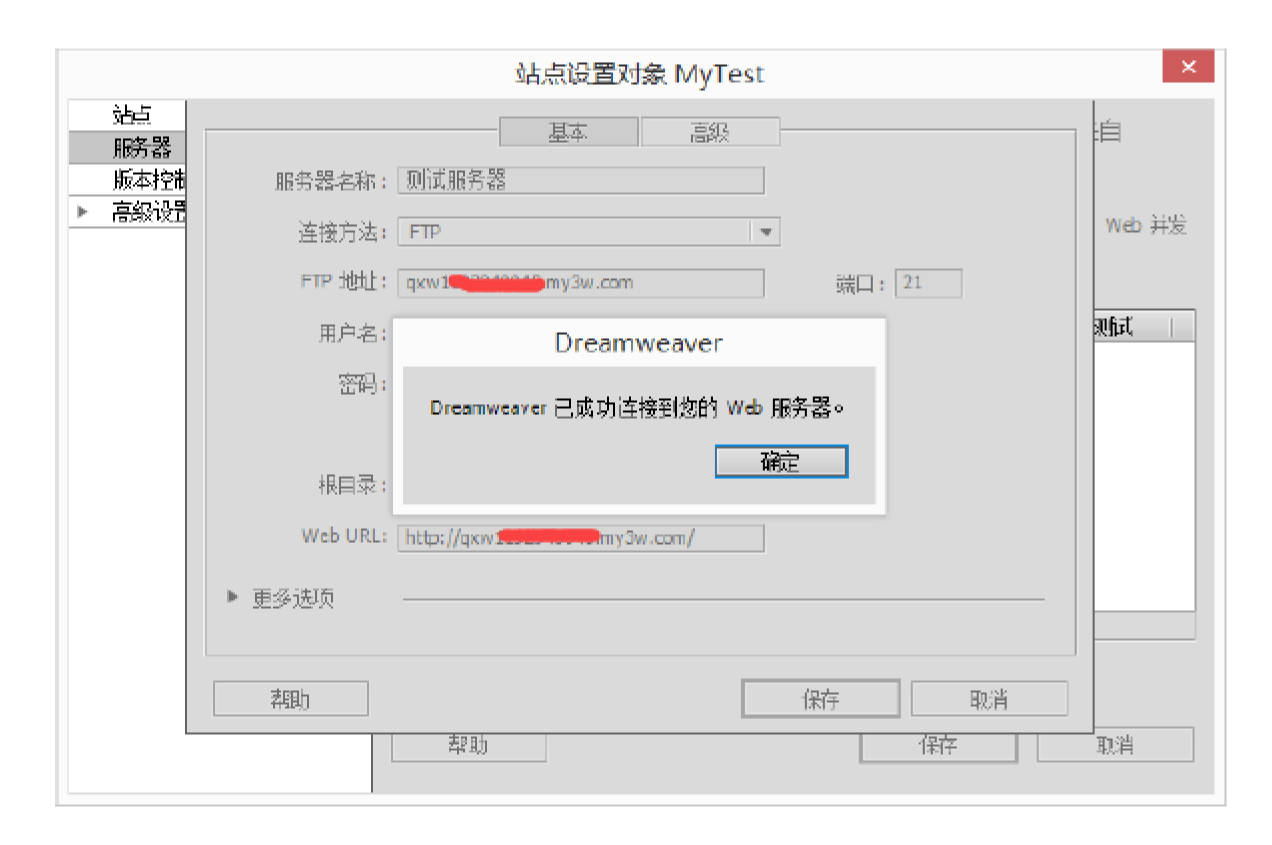


图20.5

05 单击“确定”按钮后，切换到“高级”选项卡，勾选“维护同步信息”和“保存时自动将文件上传到服务器”选项，如图20.6所示。

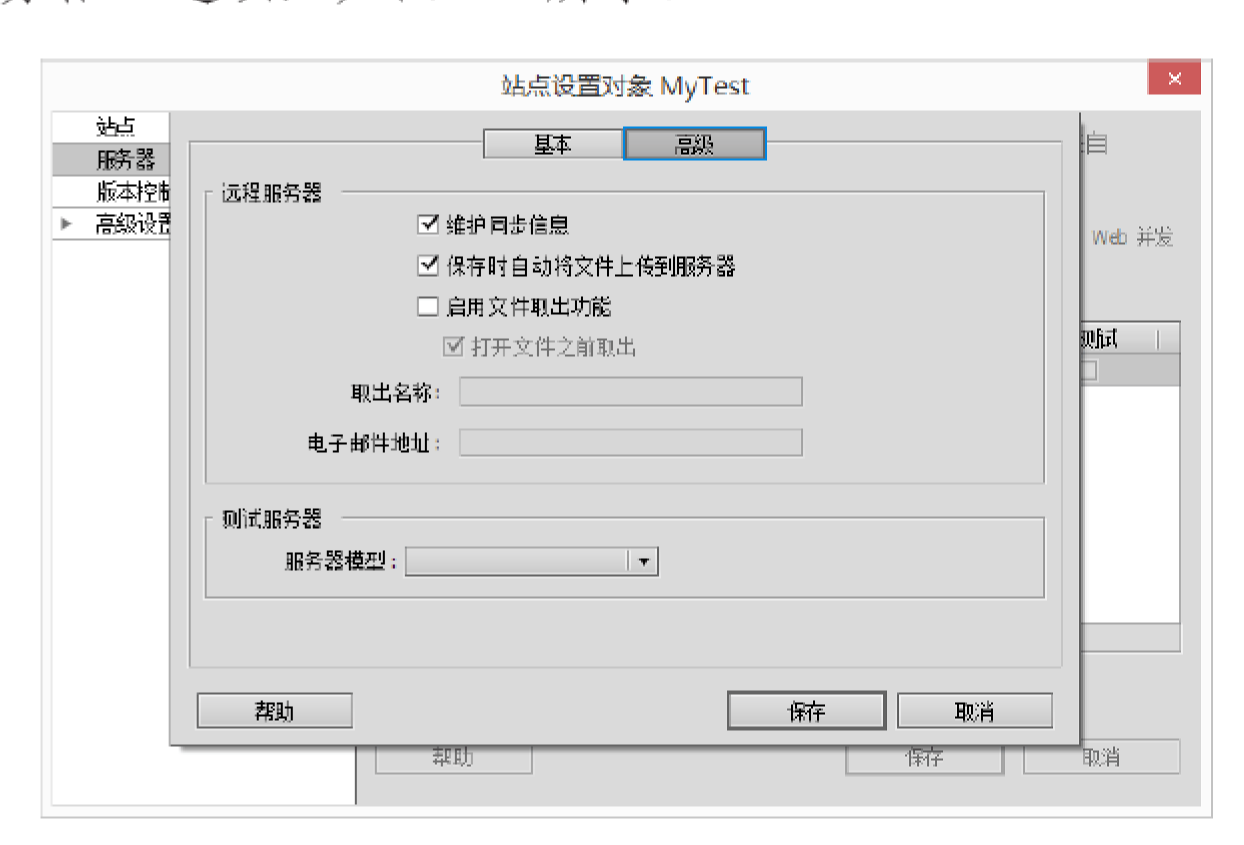


图20.6

06 单击“保存”按钮保存设置，完成新建站点的设置。在新建的站点上创建一个测试页面index.html，在页面中输入一段测试文字，当保存文件时，系统会提示您是否将文件同步保存到服务器，选择“是”即可上传文件。

07 另外，在Dreamweaver中可以选择站点，上传命令，也可以将站点下的所有文件上传到服务器。

20.2.2 使用上传工具上传

除了Dreamweaver以外，还可以使用很多FTP工具将站点上传到虚拟主机，例如flashxp、8uftp和filezilla等，下面以8uftp为例介绍如何使用上传工具上传站点。

01 启动8uftp上传工具，界面如图20.7所示。

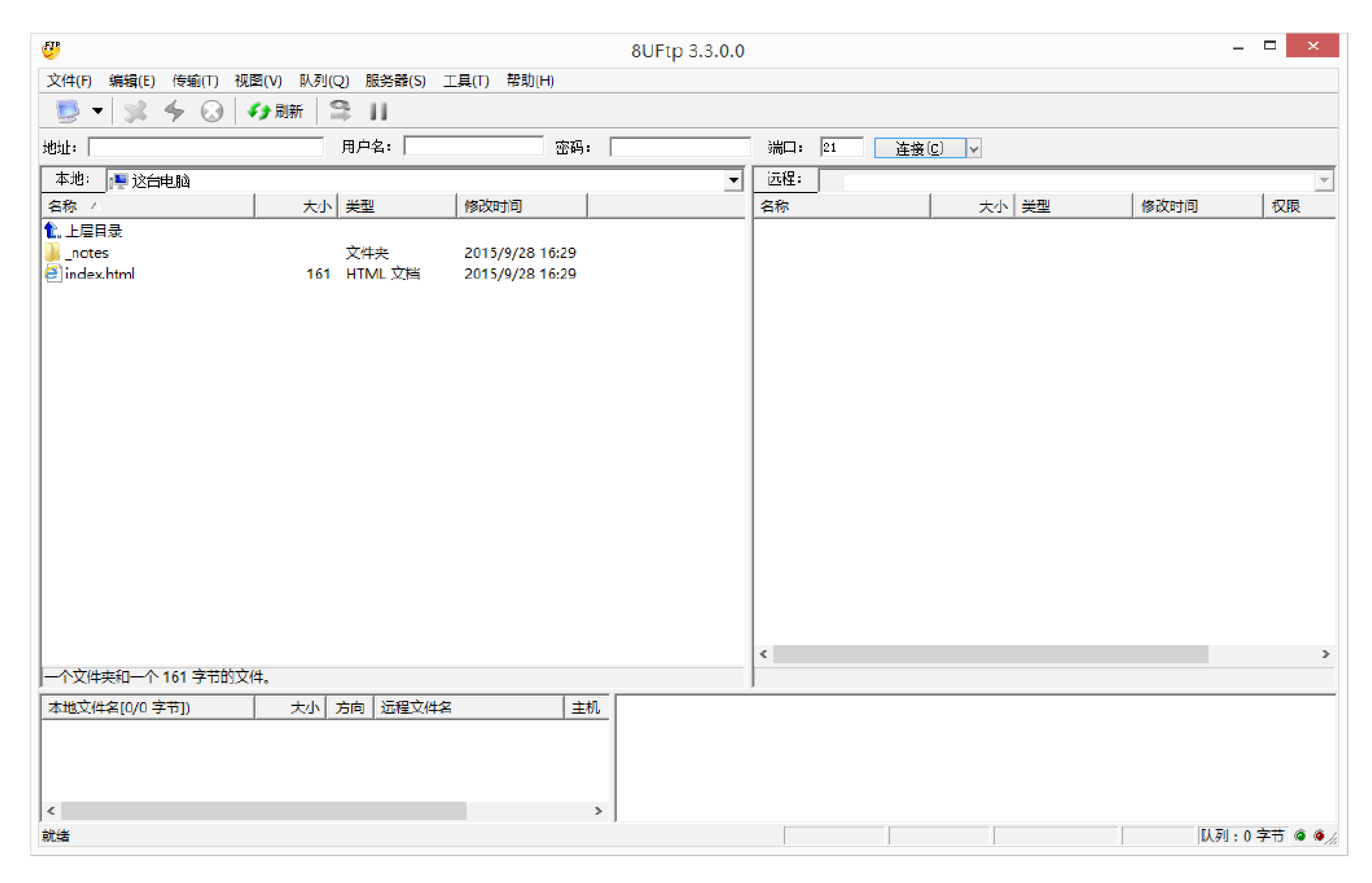


图20.7

02 选择“文件”→“站点管理器”命令，打开“站点管理器”对话框，如图20.8所示。

03 在该对话框中点击“新站点”按钮，创建一个名为MyTest的站点，并设置主机名、用户名和密码等信息，如图20.9所示。



图20.8



图20.9

04 点击“链接”按钮，将链接到服务器并返回主界面，如图20.10所示，主界面左边显示的是本地文件，右边显示的是服务器上的文件。

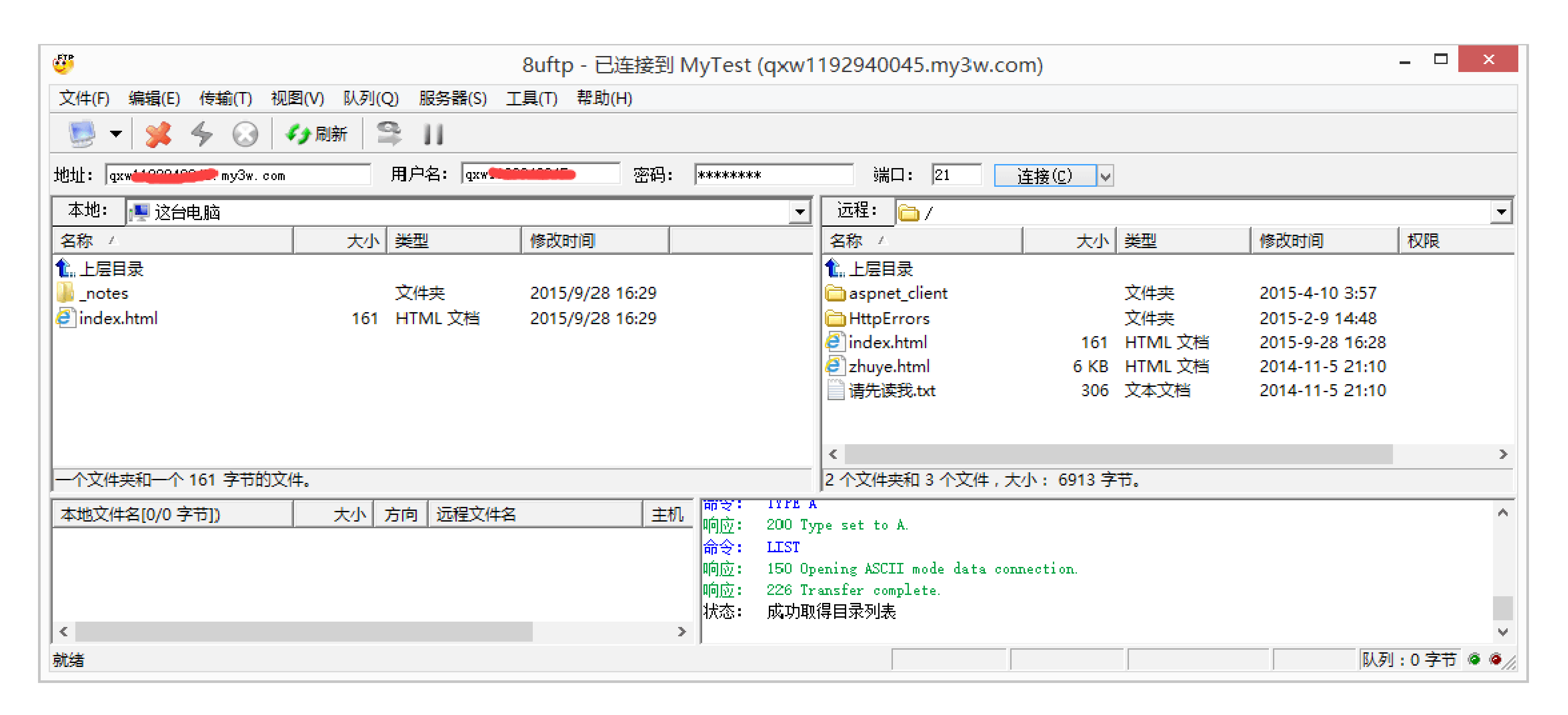


图20.10

05 将左边的文件拖动到右边，即可完成文件上传。选中左边的文件，打开右键菜单，选中上传命令也可以完成上传。如果上传多个文件，这些文件会有先后次序，在如图20.10所示左下角的区域会显示目前排队等待上传的文件，而右下角则显示当前文件上传的状态。

20.3 站点的维护与更新

创建与发布网站只是网站建设的一部分，这些工作可以经过一定努力很快完成，但是网站的维护与更新却是一个长期的工作，需要从方方面面获取信息，然后检索有用的信息，对站点进行维护，并更新网站内容，不断满足用户的需求。

20.3.1 收集与采纳用户反馈

在网站建设的时候，就需要考虑到网站后期维护的相关问题。作为经常使用网站的用户，他们会根据自己的习惯和感受，发现网站的一些弊端和问题。如果网站建设初期就考虑到为用户提供一个反馈意见的途径，当用户发现问题的时候，就可以通过这些途径向网站管理员进行反馈。

另外，网站运行过程中会产生一些不可预知的错误或问题，在建设网站的同时，有经验的程序员会根据不同情况触发信息反馈机制，当用户遇到这些问题的时候，可以通过这个机制向网站管理员提供信息。

网站上线运行初期，管理员要时刻留意网站的运行情况，对于用户反馈的比较严重的问题，要及时组织人员进行处理。另外，对于收集的用户反馈信息，也不能一次就全部接受，要有针对性的采纳，这样才能保证网站的正常运行。

20.3.2 关注用户留言

用户留言是用户反馈网站使用情况的一种方式，某些网站会为用户专门开辟一个留言版块。用户在访问网站的时候，如果遇到相关问题，或者想给网站管理员提意见，都可以从这个版块提交信息。

网站管理员要定期查看用户留言，及时了解用户使用网站的情况，掌握网站的整体运行情况，为网站的健康发展做好保障。

20.3.3 查看与回复用户邮件

基本上所有的网站导航菜单中都会有一个联系网站的菜单，进入该菜单页面后，用户会看到网站的一些联系方式，其中包括电子邮件等。某些用户习惯通过网站的电子邮件向网站管理员反馈网站的使用情况和建议。因为网站的电子邮件是网站官方对外发布的联系方式，所以网站管理员要定期查看、及时回复并掌握用户反馈的信息。

20.3.4 论坛的维护

论坛是网站运行过程中，收集用户反馈信息的最有用的功能。在论坛中可以根据网站内容划分多个模块，不同的模块安排相应的人员进行管理。可以给每个模块的管理员设置一定的权限，由他们总结各自板块中收集的用户使用信息。

目前各种论坛主要的内容就是发帖与回帖，各板块的管理员要经常关注帖子的标题与内容，对于违反相关规定的帖子，一定要及时进行处理，对于回帖中违反相关规定的内容，也不能视而不见。对于一些回帖比较多，用户比较关注的帖子，也可以将其在当前板块中置顶，或者推荐到论坛首页，这样不仅可以吸引网站的访问量，还可以了解用户对网站的需求方向。

20.3.5 站点的升级

在运行网站的过程中，为了提高安全性与用户的友好体验，都免不了要进行站点的升级。但是站点升级对于网站运行的影响也非常大，尤其影响SEO搜索排名。所以当网站运行稳定之后，对于站点升级一定要慎重。

站点升级不可避免，但是可以尽量减少升级对网站的影响，为此管理员需要注意以下几点。

（1）备份升级：我们当然可以使用Dreamweaver把网站修改的文件直接上传到服务器，但是对于一个正常访问的网站来说，并不建议这样做，因为我们不能保证每次修改的内容都能按照预期结果正确显示。所以，有必要将网站的内容在本地进行备份，修改备份的文件并进行充分的测试，最后再上传至服务器。

（2）网站基本信息：网页上的基本信息设置与搜索引擎有很大关系。如果网页的信息设置合理，搜索引擎将非常容易搜索到这些网页，如果网页的信息不定期出现变动，搜索引擎将不再收录这些网页，这对网站的搜索排名具有很大的影响。所以，网站基本信息的设置需要具有一定的前瞻性和稳定性。

（3）切勿因小失大：搜索引擎希望网站的信息尽可能的稳定，在站点升级的过程中，

如果不确定某些信息是否会影响搜索引擎，最好不要对这些内容进行更改。对于大型的网站来说，有时可能一个很小的改动，在搜索引擎看来却是巨大的变化。

20.3.6 站点内容的更新

站点的内容是网站的血与肉，一个网站如果想吸引用户、留住用户，就必须依靠网站的内容，所以网站内容的更新是关系到网站健康发展的关键。

及时更新站点内容，才能吸引更多的用户。用户希望从你的网站获取源源不断的信息，即便网站上的内容是转载的，只要满足用户的需求，就能留住用户。

网站新用户主要来源于网站推广和搜索引擎，要想搜索引擎更多的收录自己的网站，就需要不断更新网站的内容，而且搜索引擎更愿意收录原创的内容。这样，当用户在网上搜索相关内容的关键字的时候，你的网站的曝光率才会更高，才能吸引更多的新用户。



20.4 网站安全管理

网站安全一直是站长最关心的问题，一旦网站安全失守，不仅会给网站带来很多负面的影响，严重的还会泄露很多用户隐私信息。所以，网络安全管理一刻也不能放松。

20.4.1 服务器安全管理

对于个人网站而言，一般都采用虚拟主机，服务器的安全管理直接由服务提供商保证，不需要个人用户担心。

对于企业用户而言，部分用户会考虑使用自己的服务器，此时就需要考虑服务器安全管理的问题。通常情况下，企业会有自己的服务器管理人员，他们会授权相关人员使用服务器，并对相关资源进行调配和使用。管理员需要负责服务器的日常管理和维护，包括服务器账户密码的管理、主要文件的更新和修改、网站以及网站相关文件和代码的安全、漏洞的检查和管理、病毒和木马的清除等工作。

20.4.2 FTP密码的安全保护

本地文件上传到服务器的主要依据就是FTP账户和密码，任何具有FTP账户和密码的人都可以对服务器上的资源进行更改。所以，FTP密码日常的维护非常重要。可以从以下几点着手加强FTP密码的安全保护工作。

（1）口令的期限：管理员在创建临时FTP账户的时候，可以设置FTP口令的期限，比如设置口令当前有效，第二天自动消失。当下次需要创建临时FTP的时候，只需要更改一次密码即可再次使用。而对于长期使用FTP的用户，管理员可以创建一个期限比较长的FTP账户。

通过过期期限管理FTP账户密码，可以有效提高密码更改的频率。

（2）密码复杂度：很多人习惯使用生日、电话号码、身份证号码等作为密码，这就给一些坏人留下了可乘之机，为了杜绝这类事情的发生，在设置密码的时候，需要使用数字、字母和特殊符号组成的大于一定长度的密码，而且密码中必须包含大小写字母，这样可以有效提高密码的安全度。密码中也不能设置与账户相关的信息，那样也会增加密码泄露的风险。

（3）口令历史记录：如果启动了口令历史记录功能，FTP服务器会在指定的时间内禁止用户修改使用过的密码，这样可以有效提高口令的安全性。

（4）账户锁定策略：理论上说，暴力破解可以破解所有的密码。为了避免有人采用暴力破解的方式获取FTP密码，需要启动账户锁定策略。账户锁定策略可以限定用户的登录失败次数，在超过指定登录失败次数后，服务器会自动锁定这个账号，并向管理员发出警告。锁定后的账户可以通过人工解锁，也可以设置在锁定一定时间后系统自动解锁。

20.4.3 网站程序的安全管理

网站程序的安全依然是网站建设者需要考虑的重要因素之一，可以通过以下几个方面加强网站程序的安全管理。

（1）强化脚本安全：脚本是纯文本格式，恶意者通过源代码可以很容易地看到网站服务器上的数据系统库的连接用户名和密码等重要信息。进而轻松查看数据库中的所有信息，还可能篡改并造成系统损坏。因此有必要限制访问用户、设置访问权限、加密或隐藏脚本源代码。

（2）加入防注入技术：现在基于Web的攻击一般都是注入。导致注入的原因一般为对变量的过滤不完全，从而可以使入侵者非法执行程序或查询修改任意数据。现在大部分黑客入侵都是基于SQL注入实现的，所以需要在网络程序中加入防注入代码，在上传文件入口处或关键程序中增加一行代码调用防注入程序，比如URL、表单等提交信息，通过一段防止SQL注入的过滤代码即可防止出错信息暴露，或者通过转向，当系统出错时转到一个提示出错的页面等。

（3）在Web程序中应当使用Session对象：暂时或永久性地保留使用者的信息以及目前浏览网页的状态。这样有利于管理用户，防止非法用户入侵。

20.4.4 数据的安全管理

在实际的网络运行环境中，数据的备份与恢复功能是非常重要的。因为虽然可以从预防、检查、反应等方面着手减少网络信息系统的不安全因素，但是要完全保证系统不出现安全问题，这是任何人都不可能做到的。

如果数据一旦丢失，并且不能恢复，那么就可能会造成不可挽回的损失。在国外已经出现过在某个公司网络遭到损坏时，因网站建设的管理员没有保存足够的备份数据，从而无法恢复该公司的信息系统，造成无法挽回的损失，导致公司破产的先例。网络系统的数据备份策略和数据恢复方案是实现这种安全性的重要方法。

第21章 搜索引擎优化

同样类型的网站，为什么别人的网站在搜索结果中排名靠前，而自己的网站却连个影子都找不到，这就是搜索引擎优化的效果。搜索引擎优化可以让自己的网站在搜索结果中排名靠前，获取更高的曝光率，吸引更多的访问量。



21.1 搜索引擎优化概述

当今互联网发展如此的迅猛，睡一觉起来可能就有成千上万个网站诞生了，如此多的网站，搜索引擎依据什么对这些网站进行搜索排行呢？这就是搜索引擎优化需要做的事情。

21.1.1 什么是SEO

SEO是搜索引擎优化的简称，它的全称叫做Search Engine Optimization。SEO是一种利用搜索引擎的搜索规则来提高目前网站在有关搜索引擎内的自然排名的方式。用户可以从网站结构、内容建设方案、用户互动传播、页面等角度进行合理规划，使网站更适合搜索引擎的索引原则，从而获取更多的免费流量。SEO分为站外SEO和站内SEO两方面。使网站更适合搜索引擎的索引原则又被称为对搜索引擎优化，对搜索引擎优化不仅能够提高SEO的效果，还会使搜索引擎中显示的网站相关信息对用户来说更具有吸引力。

21.1.2 为什么要做SEO

如果一个网站不做SEO，那么在茫茫互联网的世界里，它将一沉到底，很难被其他用户访问。如果一个网站做了SEO，在用户搜索的时候也不一定马上会看到这个网站，那么为什么还要做SEO呢？主要是因为做了SEO后会有以下的优点。

1. 搜索流量质量高

通过其他方法推广网站，主要是把网站推到用户眼前，如果用户根本没有访问这个网站的需要，那么这对用户来说就是一种垃圾信息，会造成用户的反感。而通过SEO优化后，网站将会出现在用户搜索的结果中，这样用户会将其视为有效信息，用户访问网站的目的性更强，转化率也会更高。

2. 性价比高

SEO相对于其他网站推广方式更加经济实惠，网站只需要花费较少的投入，加上一些简

单的技术就可以获得较大的回报。这对个人站点的用户来说，性价比还是很高的。

3. 可扩展性

做SEO的关键是通过关键词的排名，以及一些长尾关键词来提高网站本身的pv，但是搜索引擎对于网站关键词的改动非常敏感，所以在改动关键词的时候不能一次性改动太多的关键词，只有慢慢添加关键词，才能不断增加自己网站的目标关键词，从而提高网站的曝光率。

4. 长期有效性

做SEO可以提高自己网站在搜索引擎中的自然排名，网站的pv会一直存在，如果网站的排名一旦做上去了，就很难再掉下来，即便有变动也不会太多。它不像其他的一些网站推广方式，如网络广告等，这些推广一旦停止付费，推广就会立即停止。做SEO的目的是将我们的网站在搜索引擎中的排名提前，增加网站的曝光率，这样才能让我们的网站排到前面去，才会吸引更多的新用户。



21.2 搜索引擎优化实战

搜索引擎优化包括很多内容，下面我们将以内部优化和外部优化两个部分来详细介绍搜索引擎优化的方法。

21.2.1 内部优化

内部优化可以理解为针对搜索引擎的算法，对网站内部结构进行调整和改变，网站内各种元素的优化都将或多或少地改变搜索引擎的收录与网站权重。

1. 页面标题优化

页面标题优化是指对<title>标签内容的优化，大多数搜索引擎都会提取网页标题的全部或部分内容作为摘要信息中的标题。因此，网页的标题必须做到主题突出、内容简洁。可以从以下几个方面优化页面标题。

（1）标题长度：搜索结果中摘要信息的标题主要来源于页面标题的内容，不同的搜索引擎会根据实际情况对标题内容进行截取。例如google的搜索结果中，摘要信息标题的长度一般在72字节（36个中文）左右，而百度则只有56字节（28个中文）左右，超出这个范围的内容将被省略。所以，在做标题优化时应尽量将有用的信息限制在28个中文字符之内。

（2）关键字分布：搜索引擎在分析页面时，在HTML源代码中自上而下进行分析，标题内容是网页中最先出现的信息。因此，在标题的最前面加上页面的主关键字，可以有效地突出页面的主题，提高相关性。

（3）关键字词频：标题中的关键词并非重复的越多越好，而是需要根据实际情况，使

主关键词和辅关键词的词频相同，而辅关键词可以是主关键词的扩展或描述。例如下面两种标题的表达方法：

```
<title>手机|5.5寸屏手机</title>
<title>手机-最新流行的5.5寸屏手机</title>
```

第1种表达方式重复了主关键词“手机”，而且还对主关键词进行了扩展“5.5寸屏手机”，而第2种表达方式对主关键词进行了描述，这样更能得到搜索引擎的青睐。

（4）关键字组合技巧：标题中的主、辅关键字可以使用竖杠（|）、空格（ ）、逗号（，）或不使用间隔的方式进行组合，这样不但可以有效拓展标题的意义，而且还可以增加页面被检索的机率。

2. 关键词标签优化

网页中的meta标签用于鉴别作者、设定页面格式、标注内容提要和关键词，以及刷新页面等等。关键词（keywords）是meta标签中非常重要的内容，对SEO优化起着至关重要的作用。对于关键词的优化，可以分为以下3个步骤。

01 关键词的选择：选择合适的关键词非常重要，可以选择品牌关键词，如某某汽车；还可以选择相关关键词，如在线小游戏；也可以选择长尾关键词，如房地产业内精英等。

02 关键词的密度：虽然搜索引擎偏爱关键词密度高的网页，但并不是关键词密度越高，搜索引擎就一定会收录。

03 关键词的锚文本：关键词加上超链接，就是一个锚文本。当互联网上大量的锚文本指向一个网站时，搜索锚文本关键词的时候，这个网站的排名将会靠前。

3. 描述标签优化

网站描述标签优化是指对description的优化，这个优化对于新的网站来说非常重要，它不仅可以为网站增加关键词密度，而且还可以吸引更多的用户。描述标签并非越长越好，而是应该控制在150个字以内，把其中的语句写成醒目的广告语，并适当的重复几次网站的目标关键词，不要写入网站地址、无关的信息以及特殊字符等。

4. 链接优化

相对来说，网站链接越多，网站的权重越高，关键字排名也会越好。另外，网站外部链接多的网站，被搜索引擎蜘蛛爬行抓取的次数和机会较多，这也是很多网站内容没有更新，快照却持续更新的原因。所以稳定的网站链接，特别是高权重外链建设，对于网站更新频率及网站排名都有作用。

6. 页面更新

在页面更新的时候，需要更新一些图文并茂的文章，因为同样权重的网站，图文并茂的网站比纯文字的网站排名要高。更新文章的时候要注意合理的段落标题结构，可以适当的把文章的关键字作为标题。

页面更新应尽量围绕网站的关键字，更新的文章标题和内容里面可以适当的包括网站的关键字，不一定是现有的关键字，可以是相同意思的关键字。

21.2.2 外部优化

网站外部优化主要是在网络环境中对网站进行的优化，主要包括外链运营和友情链接两个方面。

1. 外链运营

外链是指从别人的网站导入到自己网站的链接。导入链接所在页面的权重直接决定了我们网站在搜索引擎中的权重，所以外链对于网站优化非常重要。

外链的发布需要我们辛勤的工作，可以在各大博客、B2B平台、分类信息平台、文库、论坛等地方发布文章，内容更新围绕着自己网站的推广来做，前期不要太着急发布外链，等网站培养权重一个月后，再发布有质量的外链。

2. 友情链接

友情链接也称为交换链接，是具有一定资源互补优势的网站之间的简单合作形式，合作方可以在对方的网站上放置自己网站的logo和网站名称，并设置超链接到自己的网站上来，这也是一种互相推广的手段。

21.2.3 向搜索引擎提交网站

向搜索引擎提交自己的网站，也叫做搜索营销，目的在于让搜索引擎收录自己的网站。我们知道网站要被搜索引擎收录，除了等待搜索引擎的爬虫程序找到自己的网站，完成收录之外，还可以直接向搜索引擎提交自己的网站，达到收录的目的。

目前各搜索引擎都有自己的搜索引擎入口，下面以百度搜索引擎为例，介绍如何向搜索引擎提交自己的网站。

01 打开百度搜索引擎入口地址：http://www.baidu.com/search/url_submit.html。

02 在打开的页面中填写自己网站的链接地址，如图21.1所示。



图21.1

03 点击“提交”按钮，等待百度收录你的网站。

21.2.4 建立HTML站点地图

HTML的网站地图主要有两个作用，一个是便于浏览者了解网站的结构，另一个是为蜘蛛抓取页面信息提供方便。可以在Dreamweaver中使用锚文本的形式制作站点地图，例如下面这句：

```
<p><a href=http://blog.sina.com.cn/sitemap.html>站点地图</a></p>
```

通过制作一级和二级菜单完成站点地图的制作。创建完成后，将这个文件保存在站点根目录下，并在网站页面上设置超链接即可。

另外，还可以通过在线站点生成工具或者站点生成软件生成站点地图。对于一些页面比较多的网站而言，使用工具生成站点地图会更方便一些。